PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

Proceso de extracción de patrones secuenciales para la caracterización de fenómenos espacio-temporales

Tesis para optar el Título de Ingeniero Informático, que presenta el Bachiller:

Rodrigo Ricardo Maldonado Cadenillas

ASESOR: Ph.D. Hugo Alatrista Salas

Lima, Mayo de 2016

Resumen

El objetivo de este trabajo de fin de carrera es realizar un proceso de extracción de patrones secuenciales basado en KDD, empleando el algoritmo de minería de patrones secuenciales PrefixSpan para prever el comportamiento de fenómenos representados por eventos que cambian con el tiempo y el espacio.

Estos tipos de fenómenos son llamados fenómenos espacio-temporales, los cuales son un conjunto de eventos o hechos perceptibles por el hombre. Además, están compuestos por un componente espacial (la ubicación donde sucede el fenómeno), un componente temporal (el momento o intervalo de tiempo en el que ocurre el fenómeno) y un componente de análisis (el conjunto de características que describen el comportamiento del fenómeno).

En el mundo, se pueden observar una gran diversidad de fenómenos espacio-temporales; sin embargo, el presente trabajo de fin de carrera se centra en los fenómenos naturales, tomando como caso de prueba el fenómeno espacio-temporal de la contaminación de los ríos en Reino Unido.

Por lo tanto, con el fin de realizar un estudio completo sobre este fenómeno, se utiliza KDD (Knowledge Discovery in Databases) para la extracción del conocimiento a través de la generación de patrones novedosos y útiles dentro de esquemas sistemáticos complejos. Además, se utilizan métodos de Minería de Datos para extraer información útil a partir de grandes conjuntos de datos. Así mismo, se utilizan patrones secuenciales, los cuales son eventos frecuentes que ocurren en el tiempo y que permiten descubrir correlaciones entre eventos y revelar relaciones de “antes” y “después”.

En resumen, el presente trabajo de fin de carrera se trata de un proceso para mejorar el estudio del comportamiento de los fenómenos gracias al uso de patrones secuenciales. De esta manera, se brinda una alternativa adicional para mejorar el entendimiento de los fenómenos espacio-temporales; y a su vez, el conocimiento previo de sus factores causantes y consecuentes que se puedan desencadenar, lo cual permitiría lanzar alertas tempranas ante posibles acontecimientos atípicos.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPÍTULO 1: INTRODUCCIÓN

1.1 Presentación del problema

Se define un fenómeno espacio-temporal como un conjunto de eventos o hechos perceptible por el hombre. Este debe estar compuesto por al menos un componente espacial y otro temporal, donde el componente espacial representa la ubicación donde sucede el fenómeno, mientras que el componente temporal representa el momento o intervalo de tiempo en el que ocurre dicho fenómeno (Venkateswara y otros, 2012). Por ejemplo, en una tormenta, se puede observar que ésta sucede en un lugar (componente espacial) y en un momento dado (componente temporal). Las características que la describen tales como, la temperatura baja, la humedad media, la presión atmosférica baja, etc. pertenecen al componente de análisis (Tryfona, 1998). Así como el fenómeno anteriormente mencionado, se pueden observar una gran diversidad de fenómenos espacio-temporales. Sin embargo, el presente trabajo de fin de carrera se centra en el estudio de los fenómenos naturales tales como los terremotos, erupciones volcánicas, huracanes, epidemias, polución ambiental, entre otros. Esta elección se debe principalmente a las siguientes razones: 1) Son los fenómenos más difíciles de comprender debido a la cantidad y heterogeneidad de las características que los describen. 2) Son fenómenos que han generado un fuerte impacto en la vida del hombre y en la sociedad. Debido a lo anteriormente mencionado, existe la necesidad, en el campo científico, de construir herramientas y modelos cada vez más precisos para poder comprender estos fenómenos, saber cómo funcionan, por qué ocurren, entre otras cuestiones (U. California, 2013), de manera que se puedan proponer mejores medidas y/o acciones de prevención.

No obstante, el trabajo realizado por los científicos para el estudio de los fenómenos naturales enfrenta en la actualidad un gran número de problemas que están relacionados a la extracción de información útil a partir de datos. El primer problema es la gran cantidad de datos recolectados sobre estos fenómenos espacio-temporales. Toda esta información es trabajada de forma manual por los científicos, los cuales analizan estos datos con la finalidad de encontrar patrones sobre diversos temas específicos. El segundo problema es la diversa cantidad de atributos o variables que caracterizan a los fenómenos espacio-temporales, los cuales pueden ser de distinto tipo (números, categorías, si/no, etc.). Por ejemplo, en un terremoto, la cantidad de campos que se deben analizar sobre este fenómeno pueden llegar a ser muchos, como la profundidad, la distancia del epicentro, la

magnitud, las condiciones geológicas, entre otros. Finalmente, el tercer problema es extraer información útil a partir de estos extensos conjuntos de datos, donde realizar un análisis exhaustivo de forma manual sería inverosímil. Esto se debe a la gran cantidad de tiempo que se debe de invertir para trabajar sobre estos grandes y complejos conjuntos de datos. Es por ello que actualmente surge la necesidad de poder procesar todos estos datos con la ayuda de computadoras, de manera que se pueda mejorar la investigación de los científicos en comprender los fenómenos espacio-temporales y obtener resultados más eficientes (Fayyad, Piatetsky-Saphiro, Smyth, 2013).

Con la finalidad de poner en práctica la propuesta de solución de este trabajo de fin de carrera acerca de los problemas mencionados anteriormente, se ha decidido trabajar sobre el fenómeno espacio-temporal de la contaminación de los ríos en Reino Unido. Cabe resaltar que, este trabajo puede aplicarse sobre otros fenómenos espacio-temporales, no sólo los fenómenos naturales, debido a que lo único que se necesita es que, las bases de datos que representan los fenómenos tengan una estructura compuesta por los componentes espacial, temporal y de análisis, los cuales fueron explicados al inicio.

En este trabajo de fin de carrera, se escogió la contaminación de los ríos debido a que se trata de un fenómeno complejo de analizar para los científicos, y está relacionado a un tema relevante en la actualidad que es mejorar la calidad del agua y del medio ambiente. De esta manera, los problemas descritos en el párrafo anterior ocurren de forma similar para este fenómeno de contaminación de los ríos en Reino Unido. Es decir, los científicos deben analizar una gran cantidad de datos debido a que se deben recolectar diversas muestras de agua de río periódicamente. Dichas muestras se recolectan de múltiples estaciones de monitoreo que están establecidos a lo largo de los ríos más importantes de Reino Unido. De este modo, los científicos pueden analizar las fluctuaciones de las propiedades que determinan la calidad del agua. Algunas de estas variables son las propiedades físicas (como la temperatura), químicas (como el ph, los restos de sólidos en el agua, la demanda de oxígeno de plantas y animales, entre otros), biológicas (como la cantidad de plantas y animales, entre otros) y morfológicas de los ríos (como la forma del cauce, entre otros). Sin embargo, para los científicos estas propiedades son diversas variables que son complejas de comprender, por lo que necesitan mucho tiempo de análisis para obtener información útil. No obstante, los científicos realizan este trabajo manualmente; es decir, revisan tendencias con ayuda de la estadística y elaboran interpretaciones sobre los cambios que hayan ocurrido en el tiempo. De esta manera, tratan de explicar el comportamiento de este fenómeno con la

finalidad de poder plantear propuestas o acciones correctivas que puedan mejorar la calidad del agua de estos ríos. Es debido a todo lo anteriormente mencionado que existe la necesidad de emplear otros métodos que faciliten el análisis de datos de este fenómeno de contaminación de los ríos en Reino Unido (Gozzard, 2014).

Una de las varias maneras recientemente empleadas para el estudio de fenómenos en el ámbito de la informática es el proceso KDD (*Knowledge Discovery in Databases*), el cual es un conjunto de etapas o sub-procesos cuyo objetivo es la extracción del conocimiento a través de la generación de patrones novedosos y útiles dentro de esquemas sistemáticos complejos (Fayyad, Piatetsky-Saphiro, Smyth 1996b). Por lo tanto, emplear el proceso KDD para mejorar el estudio de los fenómenos espacio-temporales sería una buena alternativa. Además, una de las etapas dentro de este proceso es la Minería de Datos, la cual permite extraer información útil a partir de grandes conjuntos de datos a través de la extracción de patrones. De esta manera, sería posible procesar grandes volúmenes de datos y conseguir información interesante para analizar y explotar (Fayyad, Stolorz 1997). Así mismo, dentro de la Minería de datos, existe un área donde se realiza la extracción de patrones secuenciales, los cuales son eventos frecuentes que ocurren en el tiempo y que permiten descubrir correlaciones entre eventos y revelar relaciones de “antes” y “después”. Por lo tanto, extraer dichos patrones secuenciales sería una buena alternativa para poder confirmar y predecir el comportamiento de los fenómenos a través del tiempo, de manera que se puedan descubrir nuevas tendencias a partir de ello. (Han y otros 2006). De esta manera, emplear KDD y la extracción de patrones secuenciales conforman una alternativa de solución admisible para los problemas presentados anteriormente, por lo que, el presente trabajo de fin de carrera, se desarrollará siguiendo estos dos conceptos.

Después de todo lo mencionado anteriormente, es razonable plantearse la siguiente pregunta: ¿Es posible aplicar un proceso para mejorar el estudio del comportamiento de los fenómenos gracias al uso de patrones secuenciales y de esta manera prever posibles acontecimientos? Darle respuesta a esta pregunta conlleva a solucionar el problema central del presente trabajo de fin de carrera. De esta manera, el principal beneficio sobre el desarrollo del presente trabajo es que se brindará una opción adicional para mejorar el entendimiento de los fenómenos espacio-temporales; y a su vez, el conocimiento previo de sus factores causantes y consecuentes que se puedan desencadenar, lo cual permitiría lanzar alertas tempranas ante posibles fenómenos atípicos. Así, lo que se espera de este documento es contribuir a futuros trabajos sobre el uso de KDD y patrones

secuenciales para la investigación de fenómenos que cambian en el espacio y en el tiempo.

1.2 Objetivo General

El objetivo general para este trabajo de fin de carrera es: “Extraer patrones secuenciales empleando el algoritmo de Minería de Datos PrefixSpan para prever el comportamiento de fenómenos representados por eventos que cambian con el tiempo y espacialmente”.

1.3 Objetivos Específicos

Los objetivos específicos que permitirán medir el cumplimiento del objetivo general del presente trabajo son:

- Almacenar y pre-procesar los datos recolectados para mejorar la calidad y facilitar el manejo de los mismos. Además, se desarrolla un algoritmo para la transformación de la base de datos transaccional a una base de datos de secuencias (datos de entrada del algoritmo PrefixSpan).
- Emplear el algoritmo de Minería de Datos PrefixSpan para la extracción de patrones secuenciales, los cuales representan la evolución del fenómeno estudiado. Además, se realizará un estudio del desempeño del algoritmo utilizado, mediante la comparación con otros, a fin de conocer sus fortalezas.
- Interpretar los resultados obtenidos por medio de un análisis cualitativo de los mismos, gracias al análisis de la semántica de los patrones, a fin de descubrir variaciones temporales de los eventos estudiados.
- Implementar un prototipo de visualización de los patrones secuenciales, mostrando aquellos que permitan inferir algún fenómeno atípico (polución u otro) en base al análisis semántico de los patrones, mostrando además, los lugares donde ocurren.

1.4 Resultados Esperados

Los resultados esperados se presentan en la Tabla 1, donde se identifican los objetivos específicos a los que pertenecen y además se muestran los verificadores/indicadores que permiten medir a los resultados esperados. En caso no exista un verificador/indicador, entonces suponer que se trata del mismo resultado esperado.

Objetivos Específicos	Resultados Esperados	Verificadores/ Indicadores
1. Almacenar y pre-procesar los datos recolectados para mejorar la calidad y facilitar el manejo de los mismos	1.1 Base de datos con datos imputados, los cuales completan los valores perdidos en la base de datos original por medio de técnicas de limpieza de datos. Esta base de datos original de tipo espacio-temporal es la que se obtuvo a partir del sitio web data.gov.uk (ver Sección 3.3.1 para más detalle)	Porcentaje de registros perdidos
	1.2 Base de datos con datos discretizados, los cuales se definen a partir de la categorización de la base de datos con datos imputados, con la finalidad de mejorar el desempeño del algoritmo	Cantidad de atributos a discretizar Cantidad de categorías por atributo
	1.3 Base de datos transformada. Se trata de la transformación de la base de datos con datos discretizados a una base de datos de secuencias, la cual será utilizada por el algoritmo PrefixSpan	-
2 Emplear el algoritmo de Minería de Datos PrefixSpan para la extracción de patrones secuenciales, los cuales representan la evolución del fenómeno estudiado	2.1 Algoritmo PrefixSpan para la extracción de patrones secuenciales, los cuales representan la evolución del fenómeno estudiado.	Eficiencia de procesamiento y almacenamiento
	2.2 Lista de patrones generados	Cantidad de patrones generados
	2.3 Interpretación cuantitativa del desempeño del algoritmo, comparándolo con el algoritmo de	Cantidad de patrones generados

	patrones secuenciales BIDE	
3. Interpretar los resultados obtenidos por medio de un análisis cualitativo de los mismos, gracias al análisis de la semántica de los patrones, a fin de descubrir variaciones temporales de los eventos estudiados	3.1 Algoritmo de descubrimiento de los patrones que mejor describen al fenómeno. Los criterios que el algoritmo considerará son: 1) secuencias con mayor cantidad de cambios en el tiempo, 2) secuencias con la menor longitud, y 3) secuencias con la mayor cantidad de items. Así mismo, se realiza una breve interpretación de los resultados obtenidos por medio de un análisis de la semántica de los patrones, a fin de descubrir variaciones temporales de los eventos estudiados.	-
4. Implementar un prototipo de visualización de los patrones secuenciales	4.1 Programa de visualización de patrones secuenciales, mostrando aquellos patrones que mejor describen al fenómeno y los lugares donde ocurren	-

Tabla 1: Objetivos específicos, resultados esperados y verificadores/indicadores. Elaboración propia.

1.5 Herramientas, métodos y metodologías

En la presente sección, se identifican, detallan y justifican las herramientas, los métodos y la metodología que permitirán obtener los resultados descritos en la sección anterior, de manera que se cumplan los objetivos específicos planteados y por lo tanto el objetivo general de este proyecto de fin de carrera.

1.5.1 Herramientas

El siguiente cuadro muestra la relación del conjunto de resultados esperados con el conjunto de herramientas a utilizar para su obtención:

Resultado Esperado	Herramienta(s)
Base de datos con datos imputados	• Base de datos espacio-temporal de libre acceso • Sistema de administración de base de datos Postgres Enterprise Manager 5.0.1 • Orange Biolab 2.7.8
Base de datos con datos discretizados	• Orange Biolab 2.7.8
Base de datos transformada	• Sistema de administración de base de datos Postgres Enterprise Manager 5.0.1
Algoritmo seleccionado (PrefixSpan)	• Literatura que compara la performance de los algoritmos investigados en el estado del arte
Lista de patrones generados	• IDE Netbeans 8.0 • Librería SPMF para el uso del algoritmo PrefixSpan
Interpretación cuantitativa del desempeño del algoritmo	• Sistema de administración de base de datos Postgres Enterprise Manager 5.0.1
Algoritmo de descubrimiento de los patrones que mejor describen al fenómeno estudiado	• IDE Netbeans 8.0 • Literatura que contextualice el fenómeno seleccionado (contaminación de los ríos en Reino Unido) y permita descifrar el significado de los patrones obtenidos
Prototipo de visualización de patrones secuenciales	• IDE Netbeans 8.0 • PostGis 2.1.8 • QGIS Desktop 2.10.1 • API de Google Maps • JQuery 1.11.3

Tabla 2: Resultados esperados y herramientas a utilizar. Elaboración propia.

Las **bases de datos espacio-temporales de libre acceso** permiten el acceso a conjuntos de datos públicos de gran tamaño y con características temporales y

espaciales, los cuales son compartidos con la finalidad de promover la investigación de los mismos. Para este proyecto de fin de carrera se empleó una base de datos del data.gov.uk Open Up Government (ver Sección 3.3.1 para mayor detalle). El objetivo es que los datos permitan describir el fenómeno previamente escogido.

Un **sistema de administración de bases de datos** permite la carga y el manejo de los datos escogidos. Para este trabajo se utilizará Postgres Enterprise Manager 5.0.1. El objetivo de esta herramienta es que, mediante el uso de las funcionalidades de consulta, creación, modificación y eliminación de los datos, se faciliten las actividades de procesamiento de datos (PostgreSQL-es, 2010).

Orange Biolab 5.0.1 es un software gráfico de código abierto y es ampliamente utilizada para el análisis de datos haciendo uso de diversas funcionalidades. Contiene un conjunto de herramientas de pre-procesamiento, clasificación, regresión, agrupación, reglas de asociación y visualización de los datos. Por lo tanto, se trata de una herramienta que ayuda a realizar la imputación de datos faltantes, así como la discretización de la base de datos de forma eficiente y sencilla (Demšar, Curk y Erjavec, 2013).

La **literatura que compara la performance de los algoritmos** de minería de patrones secuenciales se ha obtenido de la revisión del Estado del Arte de este proyecto de fin de carrera y se utiliza como herramienta de referencia para la elección del algoritmo a usar en la extracción de los patrones secuenciales.

La utilización de la **librería SPMF de libre acceso en Java de Ph.D. Philippe Fournier-Viger**, con la finalidad de utilizar el **algoritmo PrefixSpan** como herramienta para la obtención de los patrones secuenciales (Fournier-Viger y otros, 2014). La selección de este algoritmo se realizó en base a la revisión de la literatura como se explicó previamente. Sin embargo, dicha elección también considera el interés en el estudio de la estrategia de Crecimiento de Patrones, de la cual se basa este algoritmo, la sencillez de su entendimiento y el buen rendimiento computacional que posee. Por otro lado, la selección de la librería SPMF fue debido a que es de código abierto, cuenta con una gran cantidad de colaboradores que constantemente ofrecen mejoras de implementación, y actualmente se ha vuelto muy popular entre los investigadores de ciencias de la computación debido a la facilidad de poder utilizar diversos algoritmos de minería de datos sin necesidad de volverlos a implementar (Fournier-Viger y otros, 2014).

La **literatura que contextualice el fenómeno seleccionado y permita descifrar el significado de los patrones obtenidos**, la cual, en este proyecto de fin de carrera, se obtuvo luego de una extensa investigación sobre el fenómeno seleccionado con la finalidad de que sirva como herramienta para el análisis final de

los resultados obtenidos y poder así darle un significado a los patrones secuenciales.

El uso de **Netbeans 8.0** como IDE para la utilización de la librería SPMF en Java, de manera que sea más sencilla la importación y utilización de la librería. Por otro lado, también serviría como herramienta para la implementación de dos programas empleados en la interpretación cualitativa y en el programa de visualización de los patrones secuenciales.

PostGis 2.1.8 es una extensión para la base de datos Postgres. Añade tipos de datos adicionales como los tipos geometría, geografía, mapa de bits, entre otros a la base de datos Postgres. También agrega funciones, operadores y mejoras de índices que se aplican a estos tipos espaciales. Estas funciones, operadores y tipos de datos adicionales aumentan la potencia de Postgres, convirtiéndola en una herramienta más rápida y robusta en el manejo de bases de datos espaciales (Postgis, 2015). Servirá para realizar la proyección espacial de la base de datos de ríos y que esta pueda ser utilizada en la visualización.

QGIS Desktop 2.10.1 es un Sistema de Información Geográfica (SIG) que proporciona una creciente gama de capacidades a través de diversas funciones y complementos descargables. Permite la visualización de datos vectoriales en diferentes formatos y proyecciones sin convertir a un formato interno o común. Permite componer mapas y explorar datos espaciales interactivamente con una interfaz gráfica amigable. Además, permite crear, editar y gestionar capas vectoriales en varios formatos; diseñar mapas imprimibles; entre otras funciones (QGIS, 2015). En el presente trabajo de fin de carrera, se utilizará como herramienta para la creación de los archivos que se utilizarán en la visualización de los ríos.

Finalmente, se utilizará el **API de Google Maps** y **JQuery 1.11.3.** El primero es un conjunto de herramientas y métodos que permiten la personalización de la información y visualización de mapas (W3Schools, 2015). El segundo es una librería que permite la utilización rápida y simple de varias funciones JavaScript para el desarrollo web (MDN, 2015). Ambos serán utilizados para desarrollar el programa de visualización de los patrones secuenciales.

1.5.2 Métodos y Procedimientos

El método empleado en este proyecto de fin de carrera es el Método de Discretización. Es un método que sirve para reducir el número de valores que un atributo posee mediante la agrupación de estos en un número determinado de

intervalos. Sólo se requiere especificar el número de intervalos y/o cuántos datos deberían estar incluidos en un determinado intervalo (Krzysztof y otros 2007).

La siguiente heurística suele ser usada para escoger intervalos: el número de intervalos para cada atributo no debería ser más pequeña que el número de clases a utilizar (si se las conoce). La otra heurística muy usada también se trata de escoger un número de intervalos n_{Fi}, por cada atributo, F_i (i=1,…,n) donde n es el número de atributos), así como lo siguiente:

$$n_{Fi} = M / (3* C)$$

donde M es el número de datos considerados y C es el número de categorías conocidas (Krzysztof y otros 2007).

Existen muchos algoritmos usados en este método de discretización, a continuación se describirán dos de ellos. El primero, se llama discretización de anchura equivalente y encuentra los valores mínimo y máximo de cada característica F. Luego, divide este rango en un número, $n_{Fi,}$ de intervalos especificados por el usuario y de anchura equivalente. El segundo, se llama discretización de frecuencias equivalentes y determina los valores mínimo y máximo del atributo, ordena ascendentemente todos los valores y divide el rango en un número de intervalos definidos por el usuario. En este caso, cada intervalo contiene el mismo número de valores ordenados (Krzysztof y otros 2007).

La Figura 1 muestra un ejemplo de discretización sobre los datos del atributo “X”. En la parte superior, se tiene un conjunto de datos muy continuo del atributo. Mientras que aplicando el método de discretización como se muestra en la parte inferior, se obtienen intervalos que agrupan los datos y los representan de forma más ordenada y entendible (Krzysztof y otros 2007).

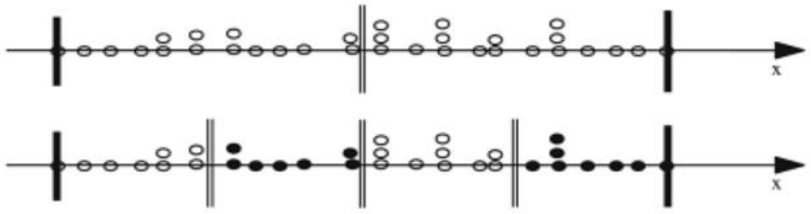

Figura 1: Ejemplo ilustrativo del método de Discretización. (Krzysztof y otros 2007).

1.5.3 Metodología

La metodología a seguir en el presente trabajo es el *Descubrimiento de Conocimiento a partir de Bases de Datos* (KDD por sus siglas en inglés), el cual será descrito con mayor detalle en la Sección 2.1.3.

1.6 Alcance

El presente trabajo de fin de carrera es un proyecto de investigación aplicada dentro del campo de la Minería de Datos, el cual forma parte del área de las Ciencias de la Computación. Tiene como fin extraer patrones secuenciales para comprender de mejor manera el comportamiento de fenómenos representados por eventos espacio temporales, empleando la metodología KDD y el uso de un algoritmo de Minería de Patrones Secuenciales. Por lo tanto, el cumplimiento de este trabajo implica la realización del proceso KDD de forma completa, abarcando la selección, pre-procesamiento y transformación de las bases de datos espacio-temporales de un fenómeno de la vida real, la utilización de un algoritmo de Minería de Patrones Secuenciales para la extracción de patrones secuenciales, la interpretación de los patrones obtenidos, y un prototipo de visualización de los resultados.

Por otro lado, las delimitaciones involucradas en el cumplimiento del presente trabajo se indican en los siguientes puntos:

- La selección de un fenómeno que pueda ser representado por datos asociados a tres componentes: un componente espacial, un componente temporal, y un componente de análisis, donde este último se compone de todas las características y propiedades que se deseen analizar del fenómeno para poder entender su comportamiento.
- La selección sólo de bases de datos transaccionales con estructura tabular, excluyendo las de otro tipo como los satelitales a través de imágenes. La exclusión mencionada se realiza debido a la mayor complejidad de manejo de los datos pues requieren mayor consumo de almacenamiento y velocidad de procesamiento. Además, la selección de bases de datos que no están disponibles para la investigación en el Perú.
- La utilización de métodos genéricos y básicos de pre-procesamiento de los datos, así como para la parte de transformación de las bases de datos. No se requiere de la utilización de métodos y herramientas sofisticadas para estas etapas, pues la heterogeneidad de los datos es mínima debido al tipo de bases de datos que se serán evaluados (bases de datos transaccionales).

- La utilización del algoritmo de minería de patrones secuenciales PrefixSpan. Básicamente se decide utilizar este algoritmo por el interés de especialización y porque obtiene resultados de forma muy eficiente en comparación a otros algoritmos bajo un paradigma fácil de comprender.
- La excepción de la etapa de validación según el proceso KDD. No se considera esta etapa del proceso ya que no se cuenta con el apoyo de un experto sobre el fenómeno seleccionado que permita validar que la interpretación de los patrones obtenidos es correcta y que apoye en la definición de ciertos parámetros dentro del proceso.

1.7 Justificación y Viabilidad

1.7.1 Justificación

El presente proyecto de fin de carrera tiene como finalidad realizar un completo estudio de los métodos de extracción de patrones secuenciales y su aplicación al estudio de fenómenos espacio-temporales. Servirá como una herramienta para mejorar el estudio del comportamiento de estos fenómenos, de manera que se permita mejorar su entendimiento y sea posible prever acontecimientos en base a los patrones generados.

Los principales beneficios que justifican la realización del presente trabajo de fin de carrera son los explicados a continuación. En primer lugar, el aporte teórico de este trabajo está en realizar un proceso genérico, completo y repetible basado en KDD para la extracción de patrones secuenciales sobre fenómenos espacio-temporales. De esta manera, se podrá realizar el mismo proceso sobre otros fenómenos espacio-temporales, consiguiendo patrones interesantes a partir de ellos que describan su comportamiento y mejoren su entendimiento. En segundo lugar, la aplicación práctica de este trabajo está en poder conseguir eficientemente patrones que representen comportamientos frecuentes del fenómeno escogido, lo que permite enfatizar el análisis sobre información relevante y útil para proponer medidas de prevención. Así mismo, al final se brinda una herramienta que permite conocer en qué lugares están ocurriendo los patrones obtenidos. Esto significa que se podrán proponer medidas de prevención aún más precisas dependiendo de las zonas en donde fueron identificados los patrones, dependiendo de la climatología, jurisdicción, geografía, etc. Además, este trabajo también permite resaltar las variaciones temporales que los atributos y/o propiedades sufren en base a los patrones secuenciales, lo cual permite que se identifiquen rápidamente dichos

cambios para poder realizar las interpretaciones necesarias que puedan explicar por qué ocurrieron esas alteraciones.

1.7.2 Viabilidad

A continuación, se presenta el estudio de viabilidad sobre los ámbitos técnico, temporal y económico para determinar si este trabajo de fin de carrera podrá ser desarrollado.

1.7.2.1 **Viabilidad técnica**

La viabilidad técnica de este proyecto se justifica debido las siguientes razones. En primer lugar, existen diversas publicaciones científicas que proveen la base teórica de este trabajo de fin de carrera, como de la metodología KDD y de la minería de patrones secuenciales, de manera que lo que se realice sea un estudio y análisis de estos documentos. En segundo lugar, existen diversas herramientas que ayudan al desarrollo de cada una de las etapas del proceso KDD, así como se detalla en la Tabla 2 de la Sección 1.5.1. Por ejemplo, las bases de datos de acceso abierto, los sistemas de administración de bases de datos y los diversos algoritmos de minería secuencial que se han desarrollado hasta la fecha (véase con más detalle en la Sección Estado del Arte) apoyarían al desarrollo de este proyecto. Finalmente, se considera la guía del asesor pues en los últimos años ha realizado importantes trabajos en el campo de minería de patrones espacio-temporales y el uso de KDD, los cuales son factores clave dentro de este trabajo.

1.7.2.2 **Viabilidad temporal**

La viabilidad temporal de este proyecto se justifica considerando una duración dos ciclos académicos universitarios, los cuales se detallan a continuación.

Se planifica tener entre cuatro y cinco meses para la investigación del tema de tesis, así como para el planteamiento de la problemática, los objetivos, las herramientas, los métodos, la metodología, el alcance, el marco conceptual y el estado del arte del presente trabajo de fin de carrera.

Por todo lado, se planifica tener entre cinco y seis meses para la implementación y elaboración de pruebas del proyecto. Se estima una mayor duración (del 50% al 60% del total del tiempo) a la etapa de pre-tratamiento de los datos (selección, pre-procesamiento y transformación de datos), debido a que se involucra una considerable cantidad de tiempo para la selección de las bases de datos, su limpieza, y su transformación a una base de datos ideal para la extracción de patrones. Al resto de etapas (minería de datos e interpretación), se les estima un

consumo del 20% del tiempo. Se considera al final dejar el 15% restante como tiempo de validación de los resultados obtenidos según los patrones generados. Finalmente, un 5% del tiempo (o más) es reservado para lanzar un plan de contingencias en el caso de que ocurra un problema técnico, metodológico u otro, debido a que el proceso KDD es un proceso iterativo, por lo que ante la presencia de errores estos deben ser corregidos considerando etapas previas del proceso.

1.7.2.3 **Viabilidad económica**

La viabilidad económica de este proyecto se justifica mediante la utilización de software libre como parte de las herramientas de desarrollo e implementación. Se planea utilizar bases de datos espacio-temporales de acceso abierto, es decir de libre disponibilidad y acceso público, así como el sistema administrador de base de datos libre Postgres Enterprise Manager 5.0.1, Orange Biolab 2.7.8 como herramienta para el pre-tratamiento de datos la cual es igualmente de libre acceso. Finalmente, se eligió Netbeans 8.0 como herramienta IDE para el desarrollo del proyecto, utilizando el lenguaje de programación Java (JDK 8).

CAPITULO 2: MARCO CONCEPTUAL Y ESTADO DEL ARTE

En el presente trabajo, se han identificado varios conceptos que requieren ser detallados pues serán usados frecuentemente durante el desarrollo de este documento. Por ello, el presente marco conceptual se organiza a través de las siguientes etapas. En primer lugar, se presenta el objetivo del marco conceptual. En segundo lugar, se describe cada concepto identificado detallándolo lo mejor posible. Por último, se finaliza a través de un resumen de la sección y una breve conclusión.

2.1 Marco Conceptual

2.1.1 Objetivo del marco conceptual

El objetivo de esta sección es brindar los conocimientos suficientes sobre la metodología KDD y cada uno de los sub-procesos que lo componen a través de la definición y el detalle de los conceptos relacionados. Todo ello, con la finalidad de permitirle al lector familiarizarse rápidamente con la metodología y técnicas a utilizar en el presente trabajo.

2.1.2 Fenómeno espacio-temporal

Se define un fenómeno espacio-temporal como un evento o hecho perceptible por el hombre que posee al menos un componente espacial y otro temporal, donde el componente espacial representa la ubicación donde sucede el fenómeno, mientras que el componente temporal representa el momento o intervalo de tiempo en el que ocurre dicho fenómeno (Venkateswara y otros, 2012).

A partir de la manifestación de un fenómeno espacio-temporal es posible conocer cuándo y dónde ocurrió dicho fenómeno, así como el conjunto de atributos y/o propiedades que permitan caracterizar al mismo (también llamado componente de análisis). Por ejemplo, en una tormenta, se puede observar que cuando ocurre dicho fenómeno, éste sucede en un lugar (componente espacial) y en un momento dado (componente temporal), donde la temperatura es baja, la humedad es media, la presión atmosférica es baja, entre otras cosas (componente de análisis) (Tryfona, 1998).

2.1.3 Descubrimiento de Conocimiento a partir de Base de Datos (KDD)

Desde 1989, el incremento de los datos digitalizados en grandes bases de datos ha significado la necesidad de extraer conocimiento a partir de extensos conjuntos de datos (Piatetsky-Saphiro 1991). Para lograr tal objetivo, KDD se ejecuta de forma

iterativa a través de una serie de pasos como se muestra en el esquema de la Figura 2 (Fayyad, Piatetsky-Saphiro, Smyth 1996b).

Se define KDD como un proceso no trivial de identificación de patrones de datos, los cuales deben ser válidos, novedosos, potencialmente útiles y comprensibles (Fayyad, Piatetsky-Saphiro, Smyth 1996c). Se dice que es un proceso no trivial pues implica mucha complejidad computacional, deben otorgar algún beneficio, y deben poder ofrecer la visualización adecuada de los resultados (Fayyad, Piatetsky-Saphiro, Smyth 1996b).

Adicionalmente, existen otras características adicionales del proceso KDD. Primero, KDD es un proceso completo de descubrimiento del conocimiento a diferencia de Minería de Datos, el cual es un paso dentro de este proceso. Segundo, es un proceso que permite la extracción de patrones o modelos, los cuales proporcionan estructuras o descripciones de alto nivel de un conjunto de datos. Finalmente, este proceso de extracción de conocimiento es fundamentalmente un trabajo estadístico. La Estadística provee un lenguaje o marco que permite cuantificar la incertidumbre que se produce cuando se trata de inferir los resultados (obtención de patrones) de una muestra particular de una población total (Fayyad, Piatetsky-Saphiro, Smyth 1996a).

Cada paso dentro del proceso KDD se definirá de forma más detallada en las siguientes secciones.

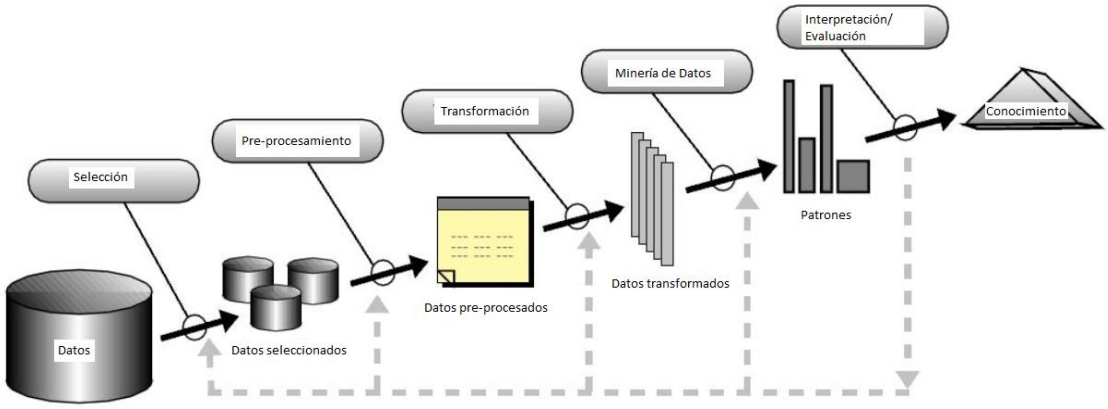

Figura 2: Esquema del proceso KDD propuesto por Fayyad. Adaptado de (Fayyad, Piatetsky-Saphiro, Smyth 1996b).

2.1.4 Selección de Datos

La selección de datos es una fase importante dentro del proceso KDD debido a que consiste en recolectar diversas fuentes de datos que guarden relación con el fenómeno a estudiar. Pueden representarse de diversas maneras como por ejemplo imágenes satelitales, registros informáticos, registros de eventos, entre otros. Por

otro lado, se debe ser prudente en la selección de los datos correctos pues deben ser lo suficientemente consistentes como conjunto, de manera que compartan características que se asocien al fenómeno en análisis (Liu, Motoda 1998). Por ejemplo, si el tema tratara de erupciones volcánicas de un determinado volcán, no sólo interesaría tener el registro de erupciones sino también otros datos como la calidad de los suelos, el registro de movimientos sísmicos cercanos, entre otros, con la finalidad extraer correlaciones de esos datos con los del fenómeno a estudiar.

2.1.5 Pre-procesamiento

Según se explica en (Arora y otros 2009), el pre-procesamiento de los datos es un paso crítico en el proceso de minería de datos, cuya función es la preparación y transformación de un conjunto de datos primitivos (*raw dataset)*.

La literatura explica que las bases de datos suelen contener muchos errores e inconsistencias. Estas presentan muchos datos que pueden considerarse irrelevantes o redundantes, pueden presentar ruido o tener tuplas inverosímiles. Es por ello que esta etapa puede tomar un tiempo de procesamiento considerable (Arora y otros 2009).

El objetivo es mejorar la calidad de los datos de manera que se facilite y vuelva más eficiente el proceso de minado. Para lograr esto, se utilizan los siguientes métodos. En primer lugar, la integración de los datos es una etapa donde se combinan los datos desde múltiples fuentes de información como bases de datos o archivos planos. El procedimiento de agregación que se emplee debe lidiar con los siguientes desafíos: (1) La identificación (ID) de las entidades, ya que un registro obtiene diferentes identificadores en distintas bases de datos. (2) La redundancia de información, donde podrían existir atributos que puedan ser usados de otra manera en otras fuentes, por ejemplo: sueldo mensual en una y sueldo anual en otra. En segundo lugar, la limpieza de datos se realiza dependiendo de los siguientes casos: (1) Para valores perdidos en la base de datos, se recomienda ignorar el registro si existen demasiados atributos faltantes o en todo caso reemplazar por el valor más probable usando alguna herramienta probabilística (árboles de decisión, entre otros); otros métodos no tan recomendables indican usar valores globales en los campos faltantes como “Desconocido” o sino rellenar los datos manualmente aunque no sea muy eficiente hacerlo de esta manera. En tercer lugar, en el caso de data ruidosa se pueden realizar las siguientes acciones: (1) Agrupamiento (*Clustering)* donde la formación de grupos (*clusters)* permite detectar valores aislados (*outliers*) que pueden ser admitidos en esos grupos. (2) Regresión

en donde se ajustan los datos a una función matemática que permita resolver el ruido a través de la predicción. Finalmente, para datos inconsistentes se requiere realizar un análisis y corrección manual ya que este tipo de problemas pudo ser causado por la falta de lógica o contracción (Arora y otros 2009).

2.1.6 Transformación

En esta etapa se busca aligerar el manejo de los datos mediante diversas técnicas. Algunas de ellas se mencionan a continuación. Primero, la transformación de datos, donde dependiendo del caso se pueden aplicar los siguientes métodos: (1) Normalización, en donde un atributo es transformado a una cierta escala, cuyo rango puede ser, por ejemplo, de -1 a 1 o 0 a 1. (2) Discretización, en donde los datos primitivos son reemplazados por conceptos más abstractos o de mayor jerarquía. Por ejemplo: en vez de tomar la edad de una persona categorizarla en joven, adulto o anciano. (3) Reducción de datos, en donde algunas estrategias que se pueden aplicar son reducción dimensional, donde atributos irrelevantes o redundantes son detectados y removidos; y reducción de numerosidad, donde los datos son reemplazados o estimados utilizando algún modelo paramétrico (Arora y otros 2009).

2.1.7 Minería de Datos

Existen múltiples definiciones en la literatura sobre minería de datos. Según (Fayyad, Stolorz 1997), minería de datos es entendido como el proceso de extracción de información a partir de bases de datos. Por otro lado, (Maimon, Rokach 2005) la define como la ciencia y tecnología de exploración de datos que permite descubrir patrones desconocidos hasta ese momento. Sin embargo, la definición más exacta para el contexto del presente trabajo es la siguiente: la minería de datos es un paso dentro del proceso KDD que consiste en aplicar análisis de datos y algoritmos de extracción, los cuales bajo ciertas limitaciones de eficiencia aceptables generen un conjunto de patrones (Fayyad, Piatetsky-Saphiro, Smyth 1996a).

Como se explica en (Fayyad, Stolorz 1997), la minería de datos tiene dos objetivos principales respecto al descubrimiento de patrones. Por un lado, la verificación, donde las hipótesis del usuario sobre el comportamiento de patrones deben ser aprobadas; y por otro lado, el descubrimiento, donde el sistema semi-autónomamente encuentra nuevos patrones.

Para lograr esto, se tiene a disposición la utilización de la siguiente lista no exhaustiva de métodos de minería de datos (Fayyad, Piatetsky-Saphiro, Smyth 1996a):

- Clasificación o *Classification*: aprender una función que mapea un dato dentro de alguna de las clases predefinidas.
- Regresión o *Regression*: aprender una función que mapea un dato a una variable de predicción de valores reales y al descubrimiento de relaciones funcionales entre variables.
- Agrupamiento o *Clustering*: identificar un conjunto de categorías que describen conjuntos de datos.
- Sumarización o *Sumarization*: encontrar una descripción compacta para un subconjunto de datos.
- Modelo de dependencia o *Model Dependency*: encontrar un modelo que describa el significado de dependencia entre variables.
- Detección del cambio y desviación o *Change and Deviation Detection*: descubrir cambios significativos en los datos a partir de valores normativos.

2.1.7.1 **Patrones Secuenciales**

Los patrones secuenciales o también llamados secuencias frecuentes son conjuntos de elementos, subsequencias o subestructuras que aparecen con frecuencia en un conjunto de datos y no sobrepasan un soporte mínimo (*threshold*) especificado por el usuario (Han y otros 2006). Tienen el objetivo de asociar y correlacionar los datos, lo cual ayuda a la indexación, clasificación, agrupamiento, entre otras tareas de minería de datos (Han y otros 2006).

Otros conceptos implicados en la definición de patrones secuenciales son los siguientes (Font 2013):

- *Item*: es un elemento cualquiera asociado a un valor literal.
- *Itemset*: sea $I = \{i_1, i_2, \ldots i_n\}$ el conjunto de todos los ítems; entonces, se define a un itemset, como un subconjunto de los ítems en *I*.
- Secuencia: es una lista ordenada de *itemsets*. Una secuencia *s* se denota como $(s_1\ s_2\ \ldots\ s_l)$, donde s_j es un itemset, es decir, $s_j \subset I$ *para* $1 \leq j \leq l$.
- Tamaño de secuencia: es la cantidad de elementos o *itemsets* en la secuencia.
- Talla de secuencia: es la cantidad de *items* en la secuencia.
- Soporte de secuencia: es la cantidad de secuencias en la base de datos que contienen la secuencia analizada.

- Soporte mínimo: es la cantidad mínima de secuencias en la base de datos que contienen la secuencia analizada.
- Secuencia frecuente: es la secuencia cuyo soporte es mayor o igual a uno especificado por el usuario.
- Subsecuencia y supersecuencia: una secuencia $\alpha = (a_1\ a_2\ \dots\ a_n)$ es llamada una subsecuencia de otra secuencia $\beta = (b_1\ b_2\ \dots\ b_m)$ y β es llamada una supersecuencia de α, si existen enteros $1 \leq j_1 < j_2 < \dots < j_n \leq m$ tal que $a_1 \subset b_{j1}$, $a_2 \subset b_{j2}$, $\dots$, $a_n \subset b_{jm}$.
- Secuencia maximal: es maximal si no es subsequencia de otra secuencia.

Un ejemplo ilustrativo es el que propuso (Agrawal, Srikant 1995), en donde se buscaba solucionar un problema de transacciones de consumidores. Dada la base de datos en la Tabla 3, se obtienen las secuencias representadas en la Tabla 4 según la cronología de consumo de cada cliente. Si el soporte mínimo es igual a dos consumidores, por ejemplo, entonces las secuencias {(30) (90)} y {(30) (40 70)} son secuencias maximales por definición y son los patrones secuenciales deseados (Tabla. 5). La primera secuencia tiene su soporte en los consumidores 1 y 4; mientras que la segunda, en los consumidores 2 y 4. Cabe mencionar que la elección del soporte mínimo dependerá de la

ID del consumidor	Tiempo de la transacción	Elementos comprados
1	Junio 25, 1993	30
1	Junio 30, 1993	90
2	Junio 10, 1993	10,20
2	Junio 15, 1993	30
2	Junio 20, 1993	40,60,70
3	Junio 25, 1993	30,50,70
4	Junio 25, 1993	30
4	Junio 30, 1993	40,70
4	Junio 25, 1993	90
5	Junio 12, 1993	90

Tabla 3: Base de datos ordenada por ID de consumidor y momento de la transacción. Adaptado de (Agrawal, Srikant 1995)

ID del consumidor	Secuencia del consumidor
1	((30) (90))
2	((10 20) (30) 40 60 70
3	((30 50 70))
4	((30) (40 70) (90))
5	((90))

Tabla 4: Secuencias obtenidas de la base de datos. Adaptado de (Agrawal, Srikant 1995)

Patrones secuenciales con soporte >= 2
((30) (90))
((30) (40 70))

Tabla 5: Patrones secuenciales obtenidos. Adaptado de (Agrawal, Srikant 1995)

2.1.7.2 Características de los Algoritmos Secuenciales

Algunas características de los algoritmos secuenciales se explican a continuación (Slimani, Lazzez 2013):

- **Escaneo Múltiple de la Base de Datos:** incluye escaneo de la base de datos original para descubrir si una larga lista de secuencias de candidatos producidos es frecuente o no.
- **Poda de Secuencia Candidato:** permite que algunos algoritmos (los de crecimiento de patrones) utilicen una estructura de datos que les permite podar secuencias candidatas al principio del proceso de minado.
- **Partición de Espacio de Búsqueda:** es un rasgo característico de los algoritmos de crecimiento de patrones. Permite la partición del espacio del espacio de búsqueda generado de largas secuencias candidatas para la gestión eficiente de la memoria.
- **Enfoque basado DFS:** con este enfoque, todos los sub-arreglos en un camino deben explorarse antes de pasar a la siguiente.
- **Enfoque basado BFS:** permite una búsqueda nivel por nivel con la finalidad de encontrar el conjunto completo de patrones (Todos los hijos de un nodo se procesan antes de pasar al siguiente nivel).
- **Restricción de la expresión regular:** tiene la propiedad llamada anti-monotonía basada en el crecimiento. Una restricción tiene esa propiedad si incluye lo siguiente característica: una secuencia debe ser alcanzable por el

crecimiento de cualquier componente que coincide con parte de la expresión regular, si satisface la restricción.

- **Búsqueda de arriba hacia abajo (top-down):** se describe que la extracción de subconjuntos de patrones secuenciales se puede hacer mediante la construcción de un conjunto de bases de datos proyectadas y extraer cada una recursivamente de arriba hacia abajo.
- **Búsqueda de abajo hacia arriba (bottom-up):** los algoritmos basados en Apriori utilizan una búsqueda bottom-up (de abajo hacia arriba), especificando cada secuencia frecuente.
- **Crecimiento Prefijo:** permite que existan las sub-secuencias frecuentes por medio de un crecimiento de prefijos frecuentes; ya que suele ser común entre un buen número de estas secuencias. Esta característica reduce la cantidad de memoria necesaria para almacenar todas las diferentes secuencias candidatas que comparten el mismo prefijo.
- **Proyección vertical de la Base de datos:** los algoritmos que utilizan esta característica visitan la base de datos secuencial sólo una o dos veces para obtener un esbozo vertical de la base de datos en lugar de la forma habitual horizontal basada en la tabla de mapa de bits o la tabla de indicación de posición construida para cada elemento frecuente.

2.1.8 Validación y Visualización de Datos

La validación se constituye como una etapa dentro del proceso KDD donde se aprueban los descubrimientos de los resultados obtenidos previamente. Se requiere del apoyo de un especialista en el área de aplicación que disponga del cocimiento y experiencia para poder confrontar los resultados. De esta manera, se confirman los patrones extraídos y se procede a la visualización de los datos.

La visualización de los datos se constituye como un paso muy importante dentro del proceso KDD pues permite facilitar el análisis, entendimiento y comunicación de los resultados obtenidos (Madalena y otros 2012). Algunos métodos de visualización de patrones secuenciales son los siguientes. Por un lado, el método de visualización gráfica representa al contenido por nodos enlazados por líneas o curvas, donde la posición de estos posee un significado en especial. Por otro lado, el método de visualización matricial muestra la información a través de tablas o matrices, donde la grilla en sí no tiene por qué mostrarse, pero sí debe respetarse la posición pues otorga significado de orden o jerarquía (Andersch 2006).

2.1.9 Conceptos relacionados al fenómeno de contaminación de los ríos en Reino Unido

En esta sección, se detallan las características que describen al fenómeno seleccionado, el cual servirá como caso de aplicación del proceso desarrollado en el presente trabajo de fin de carrera. Básicamente, se describen las propiedades físicas, químicas y biológicas del agua que componen a los ríos del Reino Unido, ya que estas características son factores relevantes que permiten determinar la calidad global del río. Aunque no se describan todas las propiedades existentes, se detallarán aquellas que serán utilizadas en el estudio del fenómeno seleccionado.

2.1.9.1 Características físicas del agua

- **Temperatura**

La temperatura del agua tiene influencia sobre las especies acuáticas debido a que pueden afectar su crecimiento y desarrollo, el éxito en reproducción, y si sobreviven o mueren. La temperatura también puede causar cambios en el consumo de oxígeno en el agua (UK Technical Advisory Group, 2008b). En la Tabla 6, UKTAG propone los estándares de temperatura para la Directiva de Marco del Agua del Reino Unido.

Tipo	Temperatura (ºC)			
	Alto	Bueno	Moderado	Malo
Agua fría	20	23	28	30
Agua caliente	25	28	30	32

Tabla 6: Estándares de temperatura. Adaptado de (UK Technical Advisory Group, 2008b)

- **Restos sólidos**

Se trata del residuo sólido que queda después de la evaporación del agua. Cuando la concentración de estos restos es excesiva, pueden reducir la penetración de la luz y causar el embotellamiento de los cauces, lo cual reduce el oxígeno en el agua, perjudicando la respiración acuática. Según lo presentado por UKTAG, la Directiva de Peces de Agua Dulce del Reino Unido especifica que el estándar para restos sólidos es una media anual de 25 mg / l (UK Technical Advisory Group, 2008b)

2.1.9.2 Características químicas del agua

- **pH**

Principalmente, la acidificación de los ríos (pH bajo) es causada por las emisiones a la atmósfera de dióxido de azufre y óxidos de nitrógeno, los cuales se oxidan para

formar ácidos que a través de lluvia o nieve se deposita en los ríos. (UK Technical Advisory Group, 2008a). La Tabla 7 muestra los estándares para el pH, propuesto por UKTAG.

pH (unidades) – Para los ríos de Inglaterra, Galés y el Norte de Irlanda			
Alto	Bueno	Medio	Malo
>=6 a <=9		4.7	4.2
pH (unidades) – Para los ríos de Escocia			
Alto	Bueno	Medio	Malo
>=6 a <=9	5.2	4.7	4.2

Tabla 7: Estándares de las condiciones de acidez. Adaptado de (UK Technical Advisory Group, 2008a)

- **Demanda biológica de oxígeno (DBO)**

DBO es una medida de la cantidad de oxígeno utilizado por la fauna y flora en el agua (VICAIRE, 2006). Es causado principalmente por los residuos de las plantas de aguas residuales y la actividad agrícola, lo que perjudica la respiración microbiana (UK Technical Advisory Group, 2008a). Los estándares propuestos por la UKTAG para la DBO se muestran en la Tabla 8.

Tipo	**Demanda biológica de oxígeno (mg/l)**			
	Alto	**Bueno**	**Medio**	**Malo**
De terreno alto y baja alcalinidad	3	4	6	7.5
De terreno bajo y alta alcalinidad	4	5	6.5	9

Tabla 8: Estándares de la demanda biológica de oxígeno. Adaptado de (UK Technical Advisory Group, 2008a)

2.1.9.3 **Características biológicas del agua**

Los organismos más importantes para las consideraciones de calidad del agua son (VICAIRE, 2006):

- En el reino animal: Crustáceos, lombrices, rotíferos, etc.
- En el reino vegetal: Plantas acuáticas enraizadas, helechos, musgos
- En el reino protista: Protozoos, algas, hongos, bacterias

2.1.10 Conclusión

En síntesis, los conceptos previamente definidos conforman la fuente de conocimientos necesarios para el desarrollo de las próximas secciones. En primera instancia, se definió un fenómeno espacio-temporal como un evento perceptible que posee un componente espacial, un componente temporal y un componente de análisis (propiedades del fenómeno). Así mismo, se describió brevemente lo investigado sobre el fenómeno seleccionado para el desarrollo de este trabajo de fin de carrera, el cual es la contaminación de los ríos en Reino Unido. Después, se definió el proceso KDD como un proceso iterativo compuesto de una serie de pasos que permiten el descubrimiento de patrones para generar conocimiento. Luego, se explicó el significado de pre-procesamiento como una etapa de preparación de datos dentro del proceso, en donde se utilizan un conjunto de métodos empleados para lograr su cometido. Después, se mencionó el concepto Minería de Datos como otro paso dentro del proceso que se encarga de la generación de patrones a través de la utilización de un algoritmo o conjunto de algoritmos, bajo ciertas limitaciones de eficiencia. Más adelante, se definió el concepto de Patrones Secuenciales como el conjunto de elementos que aparecen con cierta frecuencia y permiten correlacionar datos. Finalmente, se explicaron los conceptos de Validación y Visualización como un par de pasos importantes dentro del proceso que posibilitan el entendimiento y comunicación de los resultados obtenidos.

2.2 Estado del arte

La minería de patrones secuenciales se ha convertido en un tema muy atractivo en la última década (Han y otros 2006). En la actualidad, existe una gran cantidad de trabajos dedicados a la minería de patrones secuenciales, los cuales han permitido mejorar la eficiencia y escalabilidad de los algoritmos de minería secuencial en grandes bases de datos transaccionales (Han y otros 2006). A continuación, se realizará un análisis descriptivo de los algoritmos de minería de patrones secuenciales más usados en la actualidad, de manera que proporcionen diversas alternativas de resolución según las características específicas de cada uno ellos. Luego, se elaborará un análisis comparativo entre los algoritmos de forma que se establezcan similitudes y diferencias entre ellos. Finalmente, se concluirá lo explicado mediante una breve síntesis de lo discutido.

2.2.1 Objetivo de la revisión del estado del arte.

El estado del arte tiene por objetivo contextualizar al lector con los diversos enfoques que los principales algoritmos de Minería de Datos han adoptado hasta la

fecha y las aplicaciones que se han realizado sobre el estudio de algunos fenómenos. Haciendo uso del Método Tradicional (Silva, Menezes 2005) para la revisión de la literatura, se busca sentar discusión sobre los resultados obtenidos de manera que se llegue a una conclusión clara y objetiva de lo investigado.

2.2.2 Algoritmos de Minería Secuencial

Hasta el momento, se han realizado numerosos trabajos sobre la minería de patrones secuenciales. De la misma manera, diversas han sido las propuestas de algoritmos que han adoptado múltiples enfoques con la finalidad de conseguir resultados más precisos consumiendo cada vez menos recursos computacionales como tiempo de procesamiento y memoria. La lista siguiente no es una lista exhaustiva de todos los algoritmos existentes, pero sí menciona los más importantes hasta ahora.

2.2.2.1 Algoritmos con Base de Datos en formato Horizontal

Los algoritmos con bases de datos bajo este formato tienen la característica de que todas las k-secuencias se construyan juntas en cada k-ésima iteración del algoritmo a medida que se recorre el espacio de búsqueda. (Slimani, Lazzez 2013).

2.2.2.1.1 AprioriAll

AprioriAll (Agrawal, Srikant 1995) es uno de los primeros algoritmos introducidos dentro de la minería secuencial de patrones junto a AprioriSome y DynamicSome. Se trata de un proceso de cinco fases:

1. Fase de Ordenamiento: la base de datos original es transformada por medio del ordenamiento del conjunto de datos según el identificador del cliente como primera prioridad y el tiempo de la transacción como segunda prioridad, generando la llamada base de datos de secuencias de consumidor (Agrawal, Srikant 1995).
2. Fase *l-itemset* (*large itemset*): se encuentra el conjunto de *l-itemsets* donde cada uno cumple con el soporte mínimo dado. Luego, se realiza un mapeo mediante la asignación de un conjunto de números enteros contiguos a los *l-itemsets* con la finalidad de tratar a cada *l-itemset* como una entidad independiente (Agrawal, Srikant 1995).
3. Fase de transformación: cada secuencia de consumidor es transformada mediante la sustitución de cada transacción por el conjunto de *l-itemsets* contenidos en esa transacción. Las transacciones que no contengan *l-*

itemsets no se conservan y una secuencia de consumidor que no contiene ningún *l-itemsets* se descarta (Agrawal, Srikant 1995).

4. Fase de secuencia: trabaja sobre el conjunto de *l-itemsets* para descubrir las sub-secuencias frecuentes. Aquí es cuando se hace uso del algoritmo AprioriAll. Se inicia con un conjunto pequeño de secuencias para producir potenciales secuencias largas (candidatos) calculando el soporte de estos durante cada iteración. Aquellos que no cumplan con el soporte mínimo se podan y los que quedan se convierten en el conjunto de secuencias para la siguiente pasada. El proceso termina cuando no se generan más candidatos o estos ya no cumplen con el soporte mínimo dado (Mooney, Roddick 2013).
5. Fase maximal: encuentra todas las secuencias maximales entre el conjunto de secuencias largas (Agrawal, Srikant 1995).

2.2.2.1.2 **GSP – Generalized Sequential Patterns**

GSP (Srikant, Agrawal 1996) realiza el mismo trabajo que el algoritmo AprioriAll sin embargo no requiere encontrar todos los *itemsets* primero. Este algoritmo permite ubicar limitaciones en la separación de tiempos entre elementos adyacentes en un patrón. Además, permite que los elementos incluidos en el elemento patrón abarquen un conjunto de transacciones en una ventana de tiempo especificada por el usuario. Así mismo, permite que el descubrimiento de patrones se realice en diferentes niveles de una jerarquía definida por el usuario (Slimani, Lazzez 2013). Por otro lado, GSP descubre patrones secuenciales generalizados. El algoritmo realiza múltiples recorridos sobre la base de datos secuencial de la siguiente manera. En el primer recorrido, encuentra secuencias frecuentes que cumplen con el soporte mínimo. Luego por cada recorrido, cada secuencia de datos es examinada con el fin de modificar el número de ocurrencias (soporte) de los candidatos contenidos en esa secuencia (Slimani, Lazzez 2013).

2.2.2.1.3 **PSP**

PSP (Masseglia y otros 1998) es un algoritmo inspirado en GSP pero que propone mejoras optimizando la recuperación de datos. El proceso toma como fuente las bases de datos transaccionales y tiene el enfoque de generación de candidatos y escaneo para el descubrimiento de secuencias frecuentes. La diferencia con GSP radica en la forma en que se organizan las secuencias candidatas. GSP y sus predecesores utilizan tablas *hash* (tablas de registros e identificadores asociados) en cada nodo interno del árbol candidato, mientras que el enfoque PSP organiza los candidatos en un árbol de prefijos lo que implica una menor carga de memoria y

recuperaciones más rápidas. La estructura de árbol usada en este algoritmo sólo almacena sub-secuencias iniciales comunes a varios candidatos una vez y el nodo terminal de cualquier rama almacena el soporte de la secuencia. La suma al soporte de los candidatos se realiza por navegación a cada hoja en el árbol y luego incrementando su valor, lo que resulta más rápido que el enfoque GSP (Masseglia y otros 1998).

2.2.2.2 **Algoritmos con Base de Datos en formato Vertical**

Los algoritmos que adoptan esta característica sólo muestran un método de poda ineficaz y generan un gran número de secuencias candidatas, lo que requiere el consumo de una gran cantidad de memoria en las primeras etapas de la extracción de patrones (Slimani, Lazzez 2013)

2.2.2.2.1 **SPADE – Sequential Pattern Discovery using Equivalence classes**

El algoritmo SPADE (Zaki 2001) usa propiedades combinatorias y técnicas de búsqueda basadas en retículos que permiten el uso de restricciones sobre las secuencias halladas. Las principales características de SPADE son las siguientes. Primero, se utiliza un formato de base de datos de id-lista vertical, donde se asocia a cada secuencia una lista de objetos en el que se produce, junto con las marcas de tiempo. Se demuestra que todas las secuencias frecuentes se pueden enumerar a través de simples intersecciones temporales en id-listas. Segundo, se utiliza un enfoque teórico basado en retículos para descomponer el espacio original de la búsqueda (retículo) en piezas más pequeñas (sub-retículo) que se pueden procesar de forma independiente en la memoria principal. El enfoque general requiere tres exploraciones de base de datos, o sólo una única exploración con información pre-procesada, minimizando así los costes de entrada/salida. Tercero, disocia el problema de descomposición de la búsqueda de patrones. Se proponen diferentes estrategias de búsqueda para enumerar las secuencias frecuentes dentro de cada sub-retículo: búsqueda en amplitud (BFS) y en profundidad (DFS) (ver Sección 2.1.7.2.).

Por otro lado, los pasos principales incluyen el cálculo de 1-secuencias y 2-secuencias frecuentes, la descomposición en clases de equivalencia padre basadas en prefijos, y la enumeración de todas las demás secuencias frecuentes través BFS o DFS dentro de cada clase. (Zaki 2001).

2.2.2.2.2 **SPAM – Sequential Pattern mining using a Bitmap Representation**

SPAM (Ayres 2002) utiliza un recorrido en profundidad (DFS) del espacio de búsqueda con varios mecanismos de poda y una representación de mapa de bits vertical de la base de datos permitiendo un eficiente conteo de soporte. Se construye un mapa de bits vertical para cada elemento en la base de datos mientras la exploración de esta se realiza por primera vez con cada mapa de bits. Cada uno tiene un bit correspondiente a cada elemento de la secuencia en la base de datos. Un factor limitante potencial sobre su utilidad es su requisito de que todos los datos se ajustan en la memoria principal (Mooney, Roddick 2013).

Los candidatos se almacenan en un retículo de secuencia lexicográfico o árbol (el mismo utilizado en PSP), que les permite a los candidatos ser alargados en una de dos maneras: secuencia extendida (*Sequence Extended*) mediante el proceso de *S-step* y/o *itemset* extendido (*Itemset Extended*) a través del uso del proceso de *I-step* (ver más detalle en Ayres 2002). Estos procesos son los mismos que el enfoque adoptado en GSP y PSP, pero se llevan a cabo utilizando mapas de bits (estructura de datos binarios) para las secuencias o elementos en cuestión, aplicándoles la operación lógica AND para producir el resultado. El proceso de *S-step* requiere que un mapa de bits transformado sea creado primero mediante el establecimiento a cero de todos los bits menores o iguales al elemento en cuestión para cualquier transacción y a uno para el resto. Este mapa de bits transformado se utiliza entonces para aplicar AND con el elemento a ser adjuntado (Mooney, Roddick 2013).

El método de poda de candidatos se basa en cierre hacia abajo (*downward closure*) y se lleva a cabo tanto en candidatos *S-extension* e *I-extension* de un nodo en el árbol utilizando un BFS, lo que garantiza que todos los nodos sean visitados. Sin embargo, si el soporte de una secuencia es menor al soporte mínimo en un nodo en particular, entonces BFS no es requerido en la secuencia debido al cierre hacia abajo (Mooney, Roddick 2013).

2.2.2.2.3 **LAPIN – Last Position Induction Sequential Pattern Mining**

LAPIN (Yang y otros 2007) utiliza una lista de elementos de última posición y un conjunto de posición fronteriza de prefijo, en lugar de la proyección de árbol o generación de candidato y prueba introducidos en algoritmos previos. La principal diferencia entre LAPIN y los algoritmos anteriores es el alcance del espacio de búsqueda. Por un lado, PrefixSpan (ver detalle en 3.2.2.2) escanea la base de datos proyectada completamente para encontrar los patrones frecuentes. Por otro

lado, SPADE une temporalmente todo el *id-list* de candidatos para obtener los patrones frecuentes del siguiente sub-retículo. Por su lado, LAPIN puede obtener los mismos resultados mediante el escaneo de sólo una parte del espacio de búsqueda de PrefixSpan y SPADE, que son de hecho las posiciones de los elementos (Nizar y otros 2010).

2.2.2.2.4 **PRISM – Prime Encoding Based Sequence Mining**

PRISM (Gouda y otros 2009) es un algoritmo que utiliza un enfoque vertical para la enumeración y soporte, y que está basado en la codificación de bloque primal, el cual consiste en la teoría de factorización prima (Gouda y otros 2009).

La minería secuencial implica una enumeración combinatoria o búsqueda de secuencias frecuentes sobre el orden de secuencia parcial. PRISM utiliza el enfoque de codificación de bloques primales para representar secuencias candidatas y usa operaciones de unión sobre los bloques primales para determinar la frecuencia de cada candidato (Gouda y otros 2009).

2.2.2.3 **Algoritmos basados en Crecimiento de Patrones**

El enfoque de crecimiento patrones frecuentes elimina la necesidad de la generación candidato y poda los pasos que se dan en los algoritmos basados en Apriori, esto lo realiza mediante la compresión de la base de datos de las secuencias frecuentes en un árbol de patrones frecuentes y luego divide este árbol en un conjunto de bases de datos proyectadas, las cuales son trabajadas por separado (Han y otros 2000).

2.2.2.3.1 **FreeSpan – Frequent Pattern-Projected Sequential Pattern Mining**

FreeSpan (Pei y otros 2000) es un algoritmo que tiene el objetivo de reducir la generación de sub-secuencias candidatas. Utiliza bases de datos proyectadas para generar anotaciones de base de datos con el fin de encontrar rápidamente patrones frecuentes. La idea general de FreeSpan es utilizar elementos frecuentes para proyectar bases de datos secuenciales en un mismo conjunto de bases de datos proyectadas más pequeño de forma recursiva, utilizando los conjuntos frecuentes minados actuales y fragmentos de sub-secuencias en cada base de datos proyectada. Dos alternativas de proyección de base de datos pueden ser utilizadas: proyección Nivel por Nivel (*Level-by-level projection*) o proyección de Nivel Alternativo (*Alternate-level projection*). El método utilizado por FreeSpan divide los datos y el conjunto de patrones frecuentes a probar. Además, limita cada prueba para la correspondiente base de datos proyectada más pequeña. FreeSpan

escanea la base de datos original sólo tres veces, cualquiera que sea la longitud máxima de la secuencia.

2.2.2.3.2 **PrefixSpan – Prefix-Projected Sequential Patterns Mining**

PrefixSpan (Pei, Han y otros 2001) es un algoritmo de crecimiento de patrones que se basa en los conceptos del algoritmo FreeSpan. Sin embargo, en vez de proyectar bases de datos secuenciales basándose de cualquier sub-secuencia frecuente, PrefixSpan examina sólo sub-secuencias **prefijo** y agrega su correspondiente sub-secuencia postfija en la base de datos proyectada (Mooney, Roddick 2013).

El principal beneficio de este enfoque es la no necesidad de generar secuencias candidatas en la base de datos proyectada; es decir, PrefixSpan sólo desarrolla patrones secuenciales más largos de aquellos más cortos. Sin embargo, aunque se reduzca el espacio de búsqueda, se produce un mayor costo de eficiencia debido a la construcción de las bases de datos proyectadas (Mooney, Roddick 2013).

En la práctica, la reducción de factores puede ser significativa debido a que sólo un pequeño conjunto de patrones secuenciales crece lo suficiente en una base de datos de secuencia y por lo tanto el número de secuencias en una base de datos proyectada será más pequeño cuando los prefijos crezcan; además, será significativa porque la proyección sólo toma la porción postfija con respecto a un prefijo (Pei, Han y otros 2001).

2.2.2.3.3 **SPARSE – Sequential Pattern Mining with Restricted Search**

SPARSE (Antunes, Oliveira 2005) utiliza características híbridas entre los algoritmos basados en Apriori y de crecimiento de patrones. Combina la generación y prueba de candidatos con el espacio de búsqueda obtenido del uso de una base de datos proyectada con la finalidad de alcanzar un mejor performance en condiciones de gran densidad de patrones. Actúa iterativamente como los algoritmos basados en Apriori: después de descubrir los elementos frecuentes, busca por patrones con longitud creciente en cada paso. Termina cuando ya no existen más patrones frecuentes potenciales que buscar. La idea principal es mantener una lista de secuencias de soporte para cada candidato y verificar la existencia de soporte solo en el subconjunto de secuencias que son super-secuencias de ambos candidatos que generan, de manera similar que SPADE (Antunes, Oliveira 2005).

2.2.2.3.4 **Extensiones: BIDE – Bi Directional Extension**

BIDE (Wang, Han 2004) es un algoritmo que trabaja sobre patrones secuenciales cerrados y evita el paradigma de la generación y prueba del candidato. Poda profundamente el espacio de búsqueda y verifica eficientemente el cierre de patrón (*pattern closure*) mientras consume muy poca memoria. No necesita mantener el conjunto histórico de patrones cerrados como CloSpan, por ello escala según el número de patrones cerrados frecuentes. BIDE adopta la búsqueda a profundidad y puede presentar sus resultados de manera on-line. Además, adopta un esquema novedoso de verificación de cierre de secuencia llamado Extensión BI-direccional, y las poda el espacio de búsqueda más profundamente utilizando el método de poda *BackScan* y la técnica de optimización *Scan-Skip*. También tiene una escalabilidad lineal que se basa según el número de secuencias en la base de datos (Wang, Han 2004).

2.2.3 Aplicaciones de la Minería de Patrones Secuenciales

Han sido muchas las aplicaciones de Minería de patrones secuenciales, a continuación, se presentan algunas áreas de aplicación que han sido muy trabajadas hasta ahora.

2.2.3.1 **Aplicaciones en la Web**

El uso de la minería secuencial en este ámbito se ha desarrollado en la extracción de conocimiento e información útil del contenido y el uso de la Web (Font 2013). Muchos trabajos han abarcado diversas áreas de aplicación por ejemplo el análisis de *logs* (registros) de alguna página web (Shettar 2012), el análisis del comportamiento de acceso de los usuarios (Meiss y otros 2009) y la recomendación de páginas web (Rameshl 2011).

2.2.3.2 **Aplicaciones en Negocios**

En marketing, se han desarrollado diversas aplicaciones que estudian el comportamiento de los clientes para poder encontrar patrones y detectar mejores oportunidades de negocio. Trabajos como (Huang 2012) y (Halawani 2010) han empleado diversas técnicas que analizan los patrones de compra del cliente y su relación al tiempo, de manera que se puedan identificar los productos a ser promovidos y el público objetivo (Font 2013).

2.2.3.3 Aplicaciones en Biología

Se realiza bajo el estudio de las secuencias denominadas *motif* las cuales son aplicables a las áreas de medicina y especialmente en secuencias biológicas y genéticas para la identificación de características de las familias ADN, ARN o secuencias de proteínas, etc. (Font 2013). Un par de trabajos relacionados se desarrollaron en (Kim y otros 2011) y (Jahanian 2011)

2.2.3.4 Aplicaciones en Minería de Texto

Aquí la minería de patrones frecuentes permite la identificación de *tokens* de texto apropiados, una importante tarea de pre-procesamiento de texto que puede tener una gran influencia en los resultados de análisis entre textos, la categorización y la clasificación. Además, se aplica para el análisis de similitud entre textos, el agrupamiento de documentos, la desambiguación del sentido de palabras, la recuperación de información, la elaboración de resúmenes de textos, la atribución de autoría de un determinado texto, en la detección de plagios, entre otras (Font 2013). Dos ejemplos de trabajos sobre minería de texto son (Ledeneva 2008) y (Zhong y otros 2012)

2.2.3.5 Otras aplicaciones

Otro tipo de aplicaciones se han realizado por ejemplo en el análisis de imágenes satelitales en series de tiempo para la detección y descripción de cambios (Maruthamuthu, Kumar 2012). Otro ejemplo son los sistemas de recomendación en las redes sociales (Yu 2012).

2.2.4 Discusión

La revisión del estado del arte ha permitido discernir varios aspectos sobre la diversidad de enfoques de algoritmos de patrones secuenciales existentes. En primer lugar, los algoritmos basados en Apriori utilizan diversas estructuras interesantes para almacenar los patrones frecuentes candidatos que provinieron de la fase de generación de candidatos. En AprioriAll y GSP, se hizo uso de tablas *hash* como estructura de candidatos. En cambio, en PSP, se planteó el uso de árboles de prefijos de candidatos cuyos nodos terminales indicaban el soporte. Otro enfoque lo tuvo SPADE que hace uso de la teoría de retículos y clases de equivalencia para poder descomponer el problema original en sub-problemas más pequeños. Por el contrario, SPAM utilizó una representación de mapa de bits vertical de la base de datos permitiendo un eficiente conteo para determinar el soporte. Por otra parte, LAPIN aplica la optimización de inducción de última

posición utilizando una lista de elementos de última posición y un conjunto de posición fronteriza prefijo. Finalmente, PRISM utiliza un acercamiento vertical para la enumeración y el conteo del soporte basado en la codificación en bloques primos (teoría de factorización prima). En segundo lugar, los algoritmos basados en crecimiento de patrones se desenvuelven haciendo uso de otras estructuras que no generan candidatos pero que realizan proyecciones de la base de datos inicial. Por un lado, FreeSpan usa elementos frecuentes para proyectar bases de datos en otras más pequeñas utilizando los conjuntos frecuentes minados actuales. No obstante, PrefixSpan hace uso de otro enfoque de proyección basándose sólo en la verificación de sub-secuencias prefijas y sólo sus correspondientes sub-secuencias postfijas son proyectadas. Por otro lado, SPARSE combina la generación y prueba de candidatos con el espacio de búsqueda obtenido del uso de una base de datos proyectada con la finalidad de alcanzar un mejor performance en condiciones de gran densidad de patrones. Finalmente, los algoritmos que obtienen secuencias cerradas se han justificado no sólo debido a un resultado más compacto y completo sino a uno más eficiente (Font 2013). CloSpan utilizó un conjunto de secuencias cerradas candidatas; sin embargo, BIDE adopta un esquema novedoso de verificación de cierre de secuencia llamado Extensión BI-direccional, lo que permite evitar las limitaciones de CloSpan.

Adicionalmente, existen otras características atribuibles a cada algoritmo de patrón secuencial, cada una de ellas describe una faceta del algoritmo en cuanto a su metodología y forma de obtener patrones. A continuación, se presenta la Tabla 9 que resume y compara las características asociadas de los algoritmos.

Característica	Apriori All	GSP	PSP	SPADE	SPAM	PRISM	LAPIN	Free Span	Prefix Span
Base de Datos Estática	x	x	x	x	x	x	x	x	x
Base de Datos Incremental									
Escaneo múltiple de Base de Datos	x	x							
Poda de Secuencia Candidato		x	x	x			x		x
Partición de espacio de búsqueda	x						x		
Enfoque basado en DFS				x	x		x	x	x
Enfoque basado en BFS		x							
Restricción de expresión regular								x	x
Búsqueda top-down								x	x
Búsqueda bottom-up		x		x	x				
Crecimiento prefijo							x		x
Proyección vertical de Base de Datos				x	x	x	x		

Tabla 9: Comparación entre algoritmos basados en Apriori, crecimiento de patrones e incrementales. Tabla adaptada de (Slimani, Lazzez 2013), (Font 2013) y (Nizar y otros 2010).

Los rendimientos de eficiencia y espacio son otras características discutibles entre los algoritmos secuenciales. En la Tabla 10 se presenta un cuadro comparativo entre la velocidad y espacio utilizado por los algoritmos GSP, SPAM y PrefixSpan, considerando un tamaño de conjunto de datos determinado. De este cuadro, se observa que el mejor rendimiento en general lo tiene PrefixSpan, pero este es superado por SPAM para un tamaño de datos grande. Por otro lado, GSP se muestra como el de menor rendimiento de los tres. En cuanto al consumo de memoria, PrefixSpan supera ampliamente a GSP y SPAM. Por otro lado, el resto de algoritmos tiene como referencia de comparación a PrefixSpan en sus resultados experimentales. LAPIN sobrepasa en más de una orden de magnitud de eficiencia a

PrefixSpan en bases de datos de gran densidad (Yang y otros 2007), así también CloSpan debido al uso de restricciones que limitan el problema (Yan, Afshar 2003). BIDE, por otra parte, consume una menor cantidad de órdenes de magnitud de memoria que CloSpan, pero más de orden de magnitud de eficiencia (Wang, Han 2004).

Algoritmo	Tamaño de Conjunto de Datos	Soporte Mínimo	Tiempo de Ejecución (seg)	Uso de Memoria (MB)
GSP Apriori-based	Medio (|D| = 200 k)	Medio (1%)	2126	687
SPAM Apriori-based	Medio (|D| = 200 k)	Medio (1%)	136	574
	Largo (|D| = 800 k)	Medio (1%)	674	1052
PrefixSpan Pattern Growth	Largo (|D| = 800 k)	Medio (1%)	798	320

Tabla 10: Comparación de rendimientos entre GSP, Spam, PrefixSpan. Tabla adaptada de (Nizar y otros 2010).

En la actualidad, los métodos de patrones secuenciales han sido aplicados de diversas maneras. Algunas áreas de aplicación se han establecido en la Web, en la biología, en los negocios y otros campos de aplicación que promueven el continuo desarrollo de los patrones frecuentes. Esto debe significar lo siguiente:

a) El uso de patrones secuenciales ha significado un gran aporte a las áreas donde se aplicaron, lo que permite deducir que se obtienen resultados de mucho valor.
b) Existen múltiples fenómenos o hechos dónde aplicar minería secuencial, de manera que no sólo en las aplicaciones expuestas sino en muchas otras pueda aplicarse métodos frecuentes ya que todas las cosas están sometidas a la dimensión temporal.

2.2.5 Conclusiones sobre el estado del arte

En síntesis, la diversidad de enfoques y acercamientos sobre la obtención de patrones secuenciales ha permitido expandir la visión de alternativas algorítmicas que se pueden emplear para solucionar problemas asociadas a dinámicas

temporales. Cada algoritmo posee sus ventajas y desventajas, de manera que será importante elegir el enfoque con el que se desee trabajar para poder utilizar el algoritmo correspondiente. Por otro lado, es importante tener en cuenta las limitaciones involucradas cuando se resuelva el problema, como el poder de procesamiento y el espacio de memoria utilizado; por lo tanto, se deberá evaluar también ese rasgo dentro de las alternativas algorítmicas que se disponen. Finalmente, las aplicaciones son diversas sobre el uso de patrones secuenciales, así que es necesaria su especificación para poder solucionar el problema.

CAPÍTULO 3: PROTOCOLO DE EXPERIMENTACIÓN

En este capítulo, se especifica el fenómeno a estudiar y se explican cada una de las etapas del proceso KDD aplicadas al estudio del fenómeno seleccionado. Primero, se empieza explicando el objetivo del capítulo y la finalidad del mismo. Segundo, se describe brevemente el fenómeno a estudiar en base a lo investigado. Tercero, se explica la aplicación de la metodología KDD para el estudio del fenómeno seleccionado. Finalmente, se realiza un análisis experimental cuantitativo y cualitativo de los resultados obtenidos.

3.1 Objetivo del protocolo de experimentación

El objetivo de este capítulo es describir brevemente el fenómeno a estudiar y aplicar la metodología KDD para extraer patrones secuenciales que describan su comportamiento. La investigación realizada en la especificación del fenómeno permitirá conocer los factores que estén involucrados en el comportamiento del mismo. Luego de haber aplicado la metodología sobre los datos que representan al fenómeno, se realizará el análisis del comportamiento descrito de los patrones obtenidos. Así mismo, este capítulo tiene como finalidad culminar con los **Objetivos Específicos 1, 2 y 3** del presente trabajo de fin de carrera.

3.2 Descripción del fenómeno seleccionado

Como ya se había mencionado en la presentación del problema (ver Sección 1.1), el fenómeno que se ha seleccionado para el desarrollo de este trabajo de fin de carrera es la contaminación de los ríos en Reino Unido. Sin embargo, cabe mencionar que los métodos de extracción de patrones secuenciales son genéricos y que pueden ser aplicados a cualquier otro fenómeno que cambia en el tiempo y espacio. La selección de este fenómeno espacio-temporal se debe a la complejidad de análisis de las propiedades y características del mismo (ver Sección 1.1), y además está relacionado a un tema relevante en la actualidad que es mejorar la calidad del agua y del medio ambiente. Debido a que en Perú no se encuentran datos de libre acceso sobre este tema, se ha decido analizar datos relacionados al estudio de ríos en Reino Unido. Para más detalle sobre la descripción de este fenómeno, revisar la Sección 2.1.9, donde se detallan las características físicas, químicas y biológicas de los ríos que definen su calidad de agua, y se mencionan los estándares de calidad de agua de los ríos en Reino Unido.

3.3 Aplicación de la metodología

A continuación, se presenta la aplicación de la metodología KDD en este proyecto de fin de carrera con la finalidad de detallar las acciones realizadas en cada una de las etapas. En este caso, se trabajará sobre la base de datos del fenómeno seleccionado de contaminación de los ríos en Reino Unido; sin embargo, es posible aplicar las mismas etapas de forma similar con bases de datos de otros fenómenos espacio-temporales (considerando la existencia de los tres componentes explicados en la Sección 2.1.2). Además, será en esta sección donde se desarrolle y culminen los **Objetivos Específicos 1, 2 y 3** del presente trabajo de fin de carrera.

En el capítulo 2 (desde la Sección 2.1.3), se explicó a detalle el proceso KDD y las etapas de las que está compuesto, como se muestra en la Figura 2. Por lo tanto, haciendo un seguimiento a la metodología, se explicará cada etapa aplicada al estudio del fenómeno de la contaminación de los ríos en Reino Unido.

3.3.1 Selección de los datos

En esta etapa, se realizó una amplia investigación sobre fuentes confiables de información que sirvan como base para la experimentación en este proyecto de fin de carrera. En la Sección 2.1.2, se explica que un fenómeno espacio-temporal cuenta con los componentes temporal (cuándo ocurrió), espacial (dónde ocurrió) y de análisis (propiedades que caracterizan al fenómeno). De esta manera, la base de datos del fenómeno espacio-temporal con la que se trabaje debe contar con al menos estos tres componentes.

Para el fenómeno seleccionado en este trabajo, la base de datos que se pudo obtener representa los niveles de concentración anual de distintas medidas que determinan la calidad de los ríos más importantes en Reino Unido. Esta base de datos se pudo obtener del sitio web de acceso público data.gov.uk (Open Up Government), donde se permite el acceso a diversas bases de datos y políticas como parte de la transparencia del gobierno (Data.gov.uk, 2013). No obstante, la autoría principal le pertenece al Departamento de Medio Ambiente, Alimentación y Asuntos Rurales del Reino Unido (DEFRA).

La base de datos está en formato CSV y cuenta con un total de **709 registros**, en donde se tienen 10 regiones diferentes del estado de Reino Unido que engloban un total de 21 ríos. Por otro lado, los registros son anuales y abarcan desde el año 1980 hasta el año 2013.

La estructura de esta base de datos se muestra en la Tabla 11. Cada registro está compuesto de la región y el río (componente espacial), el año (componente

temporal), y el conjunto de características y propiedades que describen el estado de los ríos en esas dos dimensiones (componente de análisis).

Nombre de la Columna	Tipo de dato	¿Es nulo?	Descripción
ID	Int	No	Llave primaria
Región	Text	No	Región geográfica donde se tomaron los datos
Río	Text	No	Nombre del rio y localidad donde se tomaron los datos
Año	Text	No	Año en el que ocurrió la toma de datos
Temp	Text	Sí	Temperatura (°C)
pH	Text	Sí	Nivel de acidez (unidades)
Cond	Text	Sí	Conductividad del agua (µs/cm)
SS	Text	Sí	Restos de sólidos suspendidos (mg/l)
Ash	Text	Sí	Ceniza (mg/l)
DO	Text	Sí	Oxígeno disuelto (mg/l)
BOD	Text	Sí	Demanda de oxígeno biológico (mg/l)
Amm	Text	Sí	Nitrógeno amoniacal (mg/l)
Nitrite	Text	Sí	Nitrito (mg/l)
Nitrate	Text	Sí	Nitrato (mg/l)
Chloride	Text	Sí	Cloruro (mg/l)
Alkaline	Text	Sí	Alcalinidad (mg/l)
Chloroph	Text	Sí	Clorofila (µg/l)
Orthop	Text	Sí	Ortofosfato (mg/l)
AnDet	Text	Sí	Detergente aniónico (mg/l)
Latitud	Double precision	No	Coordenada espacial latitud
Longitud	Double precision	No	Coordenada espacial longitud

Tabla 11 : Estructura de la tabla “rivers”. Elaboración propia.

Para este proyecto de fin de carrera, las características con las que se van a trabajar son temperatura (temp), pH, restos de sólidos suspendidos (SS) y demanda biológica de oxígeno (BOD) debido a que son consideradas características muy influyentes en el comportamiento de la contaminación de los

ríos en Reino Unido (Uk technical advisory group, 2008a) (Uk technical advisory group, 2008b).

3.3.2 Pre-tratamiento de los datos

Esta sección corresponde a las etapas de pre-procesamiento y transformación de los datos del proceso KDD. El objetivo del pre-tratamiento es la simplificación de la variabilidad de los datos iniciales y la generación de la base de datos de secuencias como entrada a la minería de datos. Además, en esta sección se realizará el **Objetivo Específico 1** del presente trabajo de fin de carrera. Así mismo, en las subsecciones que siguen a continuación, se mencionarán las herramientas y métodos utilizados para alcanzar los resultados esperados.

3.3.2.1 Carga de la base de datos

Debido a la gran cantidad de datos y a la necesidad de poder manejarlos rápida y adecuadamente, se utilizó una base de datos local en Postgres Enterprise Manager 5.0.1. Para ello, se usó la funcionalidad de importación de esta herramienta, de manera que se pudo cargar el archivo CSV a la base de datos en la tabla “rivers”. A esta tabla se le tuvo que crear previamente la estructura apropiada, según la cantidad de campos del archivo CSV (ver Tabla 12). La Figura 3 muestra el diagrama entidad-relación de la tabla mencionada, así mismo la Figura 4 muestra la carga exitosa de los datos.

rivers		
PK	ID	int
	region	text
	rio	text
	anio	text
	temp	text
	ph	text
	cond	text
	ss	text
	ash	text
	do	text
	bod	text
	amm	text
	nitrite	text
	nitrate	text
	chloride	text
	alkaline	text
	chloroph	text
	orthop	text
	andet	text
	latitud	double precision
	longitud	double precision

Figura 3: Diagrama entidad-relación de la tabla rivers. Elaboración propia.

	id [PK] numeric	region text	river text	year text	temp text	ph text	cond text	ss text	ash text	do text
1	1	Anglian	BEDFORD OUSE - EARITH	1980	11.3556	8.2833	802.2708	19.9208	13.0667	10.4
2	2	Anglian	BEDFORD OUSE - EARITH	1981	10.5	8.1846	827.6667	16.2444	9.388	10.0
3	3	Anglian	BEDFORD OUSE - EARITH	1982	12.0192	8.1417	848.8077	20.3115		10.4
4	4	Anglian	BEDFORD OUSE - EARITH	1983	12.6458	8.1333	824.7917	18.4292	3.6	10.5
5	5	Anglian	BEDFORD OUSE - EARITH	1984	14.08	8.232	834.32	28.7		10.9
6	6	Anglian	BEDFORD OUSE - EARITH	1985	11.975	8.081	832	19.39		11.0
7	7	Anglian	BEDFORD OUSE - EARITH	1986	12.1821	8.0767	824.2	26.4929		10.4
8	8	Anglian	BEDFORD OUSE - EARITH	1987	11.3966	7.972	827.32	17.665		9.34
9	9	Anglian	BEDFORD OUSE - EARITH	1988	10.913	8	838.6087	20.1783		10.4
10	10	Anglian	BEDFORD OUSE - EARITH	1989	12.6452	8.1086	978.5714	13.9621		9.58

Figura 4: Carga de datos a la tabla rivers en Postgres. Elaboración propia.

3.3.2.2 **Limpieza de los datos**

Como se muestra en la Figura 4, algunas características, tales como “ash” (ceniza) entre otras, contienen múltiples registros con al menos un dato perdido (91% de registros con presencia de data incompleta en realidad).

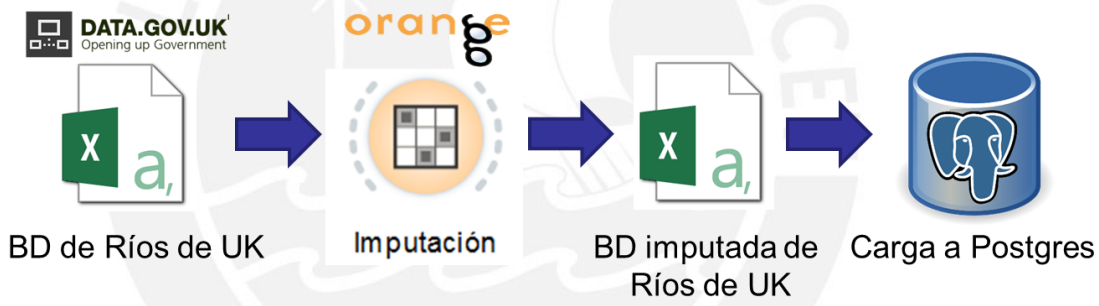

Figura 5: Proceso de Imputación de datos. Elaboración propia.

Debido a esto, como se describe en la Figura 5, utilizando la función de imputación de datos de Orange Biolab 2.7.8, se pudo realizar la limpieza de los datos perdidos en la base de datos CSV original. Para ello, los datos almacenados en formato CSV fueron tomados como parámetro de entrada de la función. Luego, se seleccionó en Orange la función “Impute”, la cual muestra una ventana que contiene las configuraciones necesarias para realizar la “limpieza” de datos.

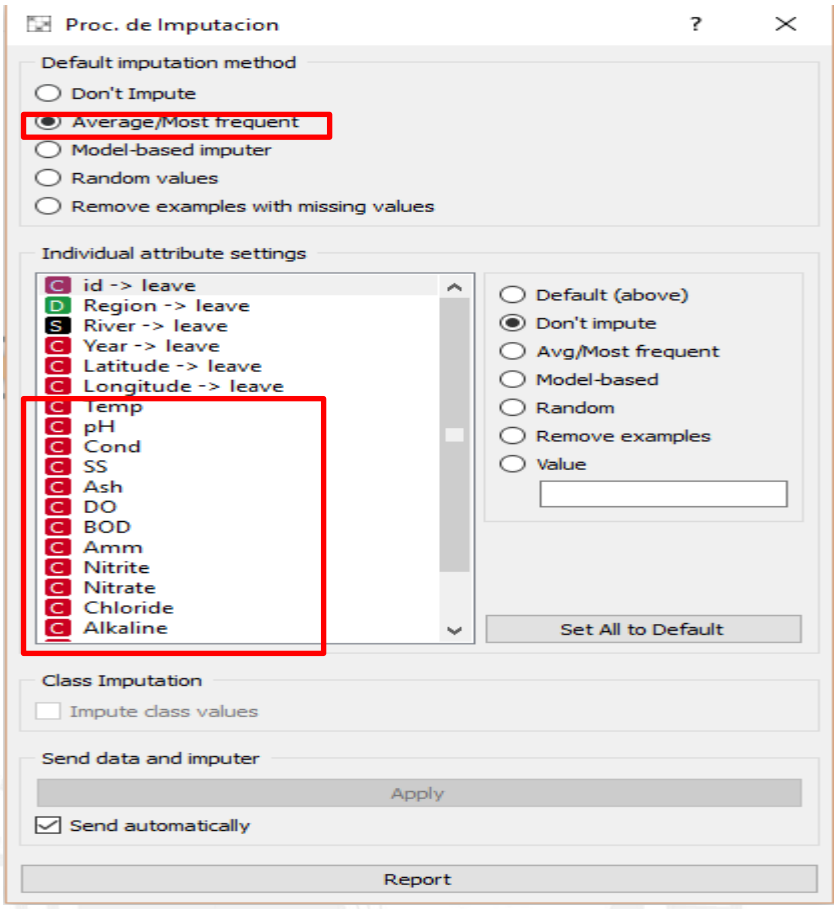

Figura 6: Ventana de configuración para la imputación de datos. Extraído de Orange Biolab 2.7.8.

Como se muestra en la Figura 6, Orange Biolab permite el uso de múltiples métodos para la imputación de datos. En este caso, se seleccionó el método de promedios o método frecuentista, el cual completa los valores faltantes utilizando el valor promedio (para atributos continuos) o el valor más frecuente (para atributos discretos). Se seleccionó este método debido a que, sobre las características escogidas para este trabajo (temp, ph, SS y BOD), sólo el 1% de los datos estaba incompleto, por lo que es viable reemplazar los valores faltantes con el promedio de los datos existentes. Finalmente, se obtuvo como salida la base de datos con datos imputados, la cual fue cargada a Postgres en la tabla “rivers”, reemplazando su contenido. De esta forma, se cumple la realización del **Resultado Esperado 1.1.**

3.3.2.3 **Discretización de los datos**

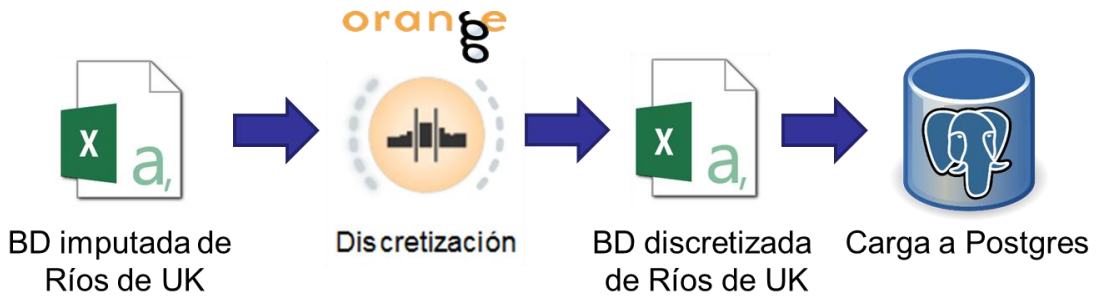

Figura 7: Proceso de discretización de datos. Elaboración propia.

A parte de la limpieza, la base de datos contiene una gran heterogeneidad de datos debido a que las características estudiadas son datos continuos (números reales). Por este motivo, como se muestra en la Figura 7, se utilizó la función de discretización de Orange Biolab 2.7.8 con la finalidad de categorizar los datos de cada característica. Para ello, se utilizó la base de datos "limpia" de la sección anterior como entrada para la función. Luego, se seleccionó en Orange la función "Discretize", la cual muestra una ventana que contiene las configuraciones necesarias para realizar la discretización de datos.

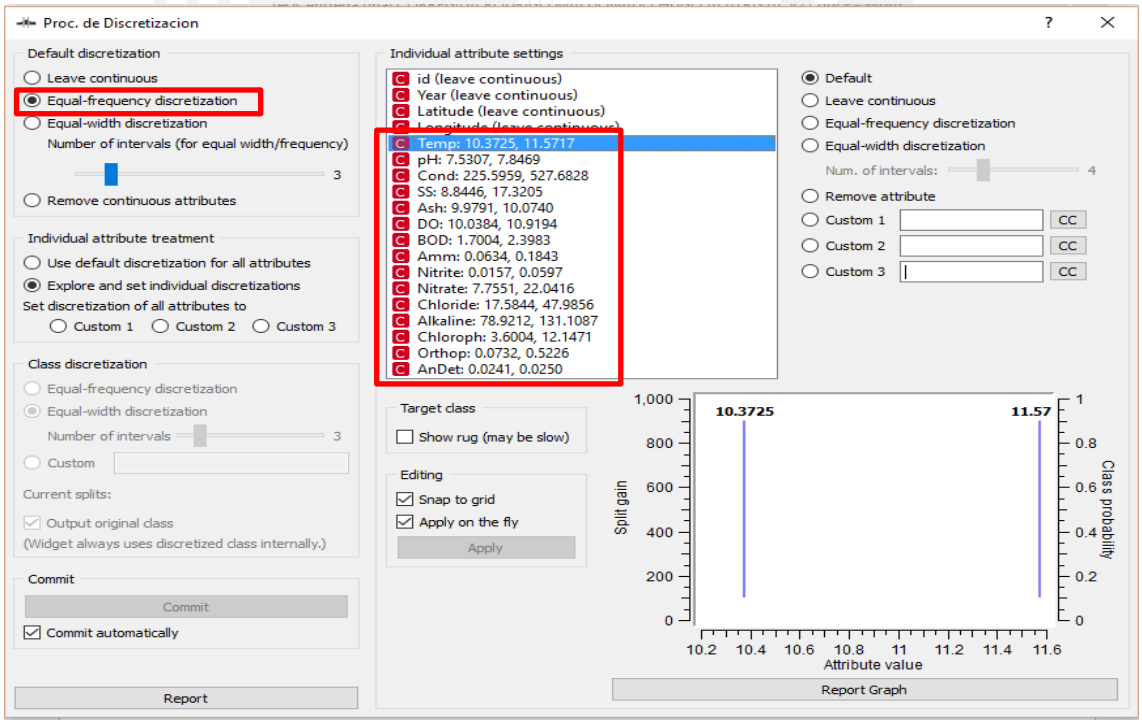

Figura 8: Ventana de configuración para la imputación de datos. Extraído de Orange Biolab 2.7.8.

Como se muestra en la Figura 8, Orange Biolab permite varios métodos para discretizar los datos. En este caso, se seleccionó el método de frecuencias equivalentes, en el cual, para cada categoría existe la misma densidad de datos (ver Sección 1.5.2). Se seleccionó este método debido a que distribuye el total de datos en grupos homogéneos de acuerdo a la frecuencia de aparición de cada dato. Por otro lado, la cantidad de categorías por cada característica se obtuvo en base a la investigación realizada sobre los estándares de las características asociadas a la calidad de agua (temp, pH, SS y BOD) (ver Sección 2.1.9). Finalmente, se obtiene como salida la base de datos con datos discretizados, la cual fue cargada a Postgres en la tabla “discretedattributes”.

La estructura de la tabla “discretedattributes” se muestra en la Tabla 12. Como se puede observar, posee la misma estructura que la tabla “rivers”; sin embargo, debido a la discretización, los valores han sido categorizados por cada característica, como se muestra en la columna “Descripción”.

Nombre de la Columna	Tipo de dato	¿Es nulo?	Descripción
ID	Int	No	Llave primaria
Región	Text	No	Región geográfica donde se tomaron los datos
Río	Text	No	Nombre del rio y localidad donde se tomaron los datos
Año	Text	No	Año en el que ocurrió la toma de datos
Temp	Text	Sí	Temperatura (°C) Bajo: <=10.3725 Medio: (10.3725,11.5717] Alto: >11.5717
pH	Text	Sí	Nivel de acidez (unidades) Bajo: <=7.5307 Medio: (7.5307,7.8469] Alto: >7.8469
Cond	Text	Sí	Conductividad del agua (µs/cm) Bajo: <=225.5959 Medio: (225.595,527.6828] Alto: >527.6828
SS	Text	Sí	Restos de sólidos suspendidos (mg/l)

			Bajo: <=8.8446 Medio: (8.8446,17.3205] Alto: >17.3205
Ash	Text	Sí	Ceniza (mg/l) Bajo: <=9.9791 Medio: (9.9791,10.0740] Alto: >10.0740
DO	Text	Sí	Oxígeno disuelto (mg/l) Bajo: <=10.0384 Medio: (10.0384,10.9194] Alto: >10.9194
BOD	Text	Sí	Demanda de oxígeno biológico (mg/l) Bajo: <=1.7004 Medio: (1.7004,2.3983] Alto: >2.3983
Amm	Text	Sí	Nitrógeno amoniacal (mg/l) Bajo: <=0.0634 Medio: (0.0634,0.1843] Alto: >0.1843
Nitrite	Text	Sí	Nitrito (mg/l) Bajo: <=0.0157 Medio: (0.0157,0.0597] Alto: >0.0597
Nitrate	Text	Sí	Nitrato (mg/l) Bajo: <=7.7551 Medio: (7.7551,22.0416] Alto: >22.0416
Chloride	Text	Sí	Cloruro (mg/l) Bajo: <=17.5844 Medio: (17.5844,47.9856] Alto: >47.9856
Alkaline	Text	Sí	Alcalinidad (mg/l) Bajo: <=78.9212 Medio: (78.9212,131.1087] Alto: >131.1087
Chloroph	Text	Sí	Clorofila (µg/l)

			Bajo: <=3.6004 Medio: (3.6004,12.1471] Alto: >12.1471
Orthop	Text	Sí	Ortofosfato (mg/l) Bajo: <=0.0732 Medio: (0.0732,0.5226] Alto: >0.5226
AnDet	Text	Sí	Detergente aniónico (mg/l) Bajo: <=0.0241 Medio: (0.0241,0.0250] Alto: >0.0250
Latitud	Double precision	No	Coordenada espacial latitud
Longitud	Double precision	No	Coordenada espacial longitud

Tabla 12: Estructura de la tabla “discrettedatributes”. Elaboración propia.

La Figura 9 muestra los datos discretizados cargados en la tabla “discretedattributes” en la base de datos Postgres. De esta forma, se cumple la realización del **Resultado Esperado 1.2.**

	id [PK] numeric	region text	river text	year text	temp text	ph text	ss text	nitrate text
1	1	Anglian	BEDFORD OUSE - EARITH	1980	(10.9505, 11.8837]	>7.8469	>17.3205	>25.4727
2	2	Anglian	BEDFORD OUSE - EARITH	1981	(10.0112, 10.9505]	>7.8469	(8.8446, 17.3205]	>25.4727
3	3	Anglian	BEDFORD OUSE - EARITH	1982	>11.8837	>7.8469	>17.3205	>25.4727
4	4	Anglian	BEDFORD OUSE - EARITH	1983	>11.8837	>7.8469	>17.3205	>25.4727
5	5	Anglian	BEDFORD OUSE - EARITH	1984	>11.8837	>7.8469	>17.3205	>25.4727
6	6	Anglian	BEDFORD OUSE - EARITH	1985	>11.8837	>7.8469	>17.3205	>25.4727
7	7	Anglian	BEDFORD OUSE - EARITH	1986	>11.8837	>7.8469	>17.3205	>25.4727
8	8	Anglian	BEDFORD OUSE - EARITH	1987	(10.9505, 11.8837]	>7.8469	>17.3205	>25.4727
9	9	Anglian	BEDFORD OUSE - EARITH	1988	(10.0112, 10.9505]	>7.8469	>17.3205	>25.4727
10	10	Anglian	BEDFORD OUSE - EARITH	1989	>11.8837	>7.8469	(8.8446, 17.3205]	>25.4727

Figura 9: Carga de datos discretizada a la tabla discretedattributes en Postgres. Elaboración propia.

3.3.2.4 **Preparación de los datos discretizados para la Minería de datos**

Con la finalidad de preparar la base de datos discretizada de la sección anterior al formato requerido por el algoritmo PrefixSpan para la etapa de minería de datos, fue necesario realizar una transformación de los datos discretizados a una base de datos de secuencias, debido a que el algoritmo PrefixSpan procesa esos tipos de

datos. Esto se realizó mediante la implementación de una función en PL/pgSQL en Postgres Enterprise Manager 5.0.1.

Previamente a la transformación, se necesitaba agregar un prefijo a los valores de las características de la tabla “discrettedattributes”, con la finalidad de poder identificar dichas características en la etapa de minería de datos. Un ejemplo del etiquetado es colocar el prefijo “temp_” en sus categorías correspondientes, obteniendo temp_<=10.3725, temp_(10.3725,11.5717] y temp_>11.5717 en este caso. Así mismo, era necesario que en cada característica no existieran espacios en blanco, debido a que posteriormente este carácter sería utilizado con otra finalidad. Por lo tanto, se ejecutaron scripts para el etiquetado de características y la limpieza de espacios en blanco (ver Anexo 1).

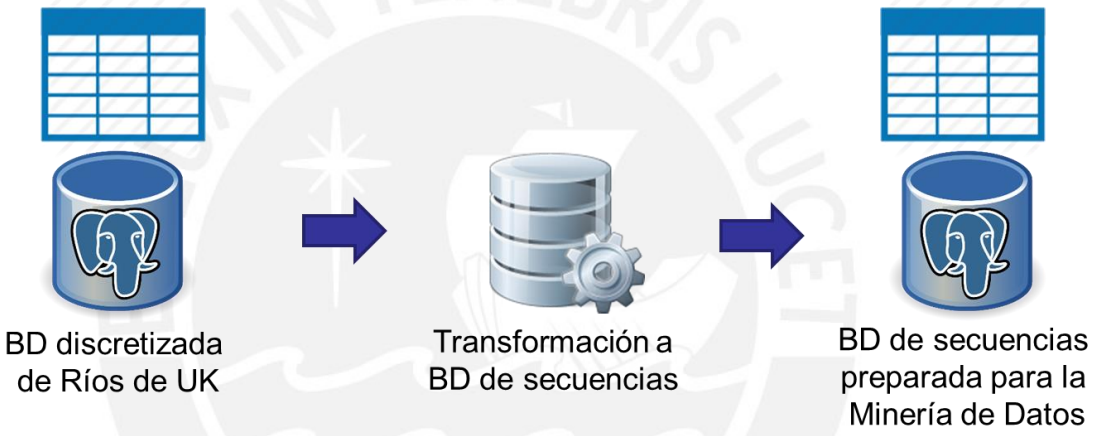

Figura 10: Proceso de transformación a la base de datos de secuencias.
Elaboración propia.

Luego, respecto a la función de transformación, se realizó el proceso mostrado en la Figura 10, donde se toma como entrada la base de datos con datos discretizados obtenidos en la sección anterior (bdd) y se la transforma a una base de datos de secuencias. El pseudocódigo de la función se muestra en el Algoritmo 1.

Inicio de función de transformación (bdd)

1. Inicializar variables zonaAnterior = ‘’, tiempoAnterior = ‘’, zonaActual = ‘’, tiempoActual = ‘’, flag = 0 y cadenaSecuencias=’’
2. Ordenar ascendentemente bdd por zonas y por tiempo
3. Para (cada registro reg en bdd)
    a. Si flag == 1
        i. zonaAnterior = reg.zona
        ii. tiempoAnterior = reg.tiempo

iii. flag = 0

b. zonaActual = reg.zona

c. tiempoActual = reg.tiempo

d. Si (zonaAnterior == zonaActual)

i. Si (tiempoAnterior == tiempoActual)

1. Concatenar en cadenaSecuencias las características del registro reg separadas por espacios

ii. Sino

1. Concatenar en cadenaSecuencias el separador de itemsets “-1”

2. Concatenar en cadenaSecuencias las características del registro reg separadas por espacios

e. Sino

i. Concatenar en cadenaSecuencias el separador de secuencias “-1 -2”

ii. Calcular the_geom

iii. Insertar cadenaSecuencias y the_geom a tabla de secuencias listofsequences

iv. Inicializar cadenaSecuencias = ’’

v. Concatenar en cadenaSecuencias las características del registro i separadas por espacios

f. zonaAnterior = zonaActual

g. tiempoAnterior = tiempoActual

//insertar la secuencia de la última zona

4. Concatenar en cadenaSecuencias el separador de secuencias “-1 -2”

5. Calcular the_geom

6. Insertar cadenaSecuencias y the_geom a tabla de secuencias listofsequences

Fin función de transformación

Algoritmo 1: Función de transformación a la base de datos de secuencias. Elaboración propia.

Para poder construir dichas secuencias, se recuerdan las definiciones de la Sección 2.1.7.1.

- *Item*: es un elemento cualquiera asociado a un valor literal. En este caso, corresponde a **los valores** de las características del fenómeno seleccionado; es decir, las categorías de la base de datos con datos

discretizados. Por ejemplo, temp_<=10.3725, temp_(10.3725,11.5717] ó temp_>11.5717 que son los valores de la temperatura; ph_<=7.5307, ph_(7.5307,7.8469] o ph_>7.8469 que son los valores de pH; etc.

- *Itemset*: es una estructura compuesta por un item o una **lista** de items. Un itemset representa una ocurrencia temporal del fenómeno, la cual está compuesta de las características que sucedieron en dicho momento sobre el fenómeno. Cada ocurrencia temporal suele representarse dentro de un paréntesis, agrupando los items. Por ejemplo, en la fecha 1, se registró sólo la temperatura (temp_<=10.3725); en la fecha 2, se registraron la temperatura y pH respectivamente (temp_(10.3725,11.5717] ph_>7.8469); en la fecha 3, se registraron la temperatura y el pH respectivamente (temp_>11.5717 ph_>7.8469); etc.
- Secuencia: es una lista de itemsets ordenados cronológicamente, los cuales ocurren en una misma entidad espacial (un río, una ciudad, etc.). Cada secuencia suele representar dentro de un paréntesis, agrupando los itemsets. Por ejemplo, en la zona x, la siguiente secuencia ((temp_<=10.3725) ((temp_10.3725,11.5717] ph_>7.8469) (temp_>11.5717 ph_>7.8469)) representa los sucesos en el tiempo que describen la temperatura y pH del fenómeno.

Sin embargo, debido a que en la etapa de minería de datos se utilizará la librería de acceso abierto SPMF descrita en la Sección 1.5.1, se consideró el formato de las secuencias requerido por esta librería como parte de la implementación de la función. Un ejemplo sencillo del formato requerido se describe en la Tabla 13. En este caso, cada ítem se separa por un espacio en blanco, cada itemset se separa con “-1” (línea 3.d.ii.1 del Algoritmo 1), y cada secuencia se separa con “-1 -2” (líneas 3.e.i y 4 del Algoritmo 1). Es por ello que previamente los espacios en blanco fueron removidos.

Secuencias
temp_(10.3582,11.5801] ph_>7.8469 bod_>17.3205 ss_>22.1594 -1 temp_(10.3582,11.5801] ph_>7.8469 bod_(8.8446,17.3205] ss_>22.1594 -1… -1 -2
temp_<=10.3582 ph_(7.5307,7.8469] bod_>17.3205 ss_>22.1594 -1 temp_<=10.3582 ph_>7.8469 bod_>17.3205 ss_>22.1594 -1 … -1 -2
temp _>11.5801 ph_(7.5307,7.8469] bod_>17.3205 ss_>22.1594 -1

temp_(10.3582,11.5801] ph_(7.5307,7.8469] bod_>17.3205 ss_>22.1594 -1 … -1 -2
…

Tabla 13: Formato de la base de datos de secuencias requerido por SPMF, como entrada para ejecutar el algoritmo PrefixSpan. Elaboración propia.

Además, la función implementada también contempló el cálculo de los puntos geométricos espaciales en la columna the_geom utilizando la función espacial ST_SetSRID (ST_MakePoint ("longitud", "latitud"), 4326)) de PostGis 2.1.8 (líneas 3.e.ii y 5 del Algoritmo 1). Como la proyección espacial utilizada por Google es el sistema de proyección espacial Web Mercator, donde el Sistema Geodésico Mundial de 1984 (WSG 84 o también llamada EPGS: 4326) es el estándar usado en la cartografía y navegación (Google Developers, 2015), entonces se usó este sistema. Este atributo the_geom será posteriormente utilizado en el capítulo 4, cuando se explica la visualización de los datos y ríos.

Finalmente, la función implementada realiza la carga de las secuencias a la base de datos en la tabla "listofsequences" (líneas 3.e.iii y 6 del Algoritmo 1), la cual posee la estructura mostrada en la Tabla 14. De esta forma, se cumple la realización del **Resultado Esperado 1.3.**

Nombre de la Columna	Tipo de dato	¿Es nulo?	Descripción
ID	Int	No	Llave primaria
River_name	Text	No	Región geográfica y nombre del río donde se tomaron los datos
Sequences	Text	No	Secuencia construida a partir de la función implementada (Anexo 2)
Latitud	Double precision	No	Coordenada espacial latitud
Longitud	Double precision	No	Coordenada espacial longitud
The_geom	Geometry	No	Punto geométrico en base a la latitud, longitud, y proyección espacial EPGS: 4326

Tabla 14: Estructura de la tabla listofsequences. Elaboración propia.

3.3.3 Minería de Datos e Interpretación de los resultados

Esta sección tiene por objetivo generar los patrones o secuencias frecuentes que permitan resumir el comportamiento del fenómeno escogido y realizar la interpretación de los mismos. Así mismo, en esta sección se desarrollan los **Objetivos Específicos 2 y 3** del presente trabajo de fin de carrera.

Por este motivo, se toma como entrada las secuencias construidas en la etapa de pre-tratamiento de datos.

3.3.3.1 **Minería de datos**

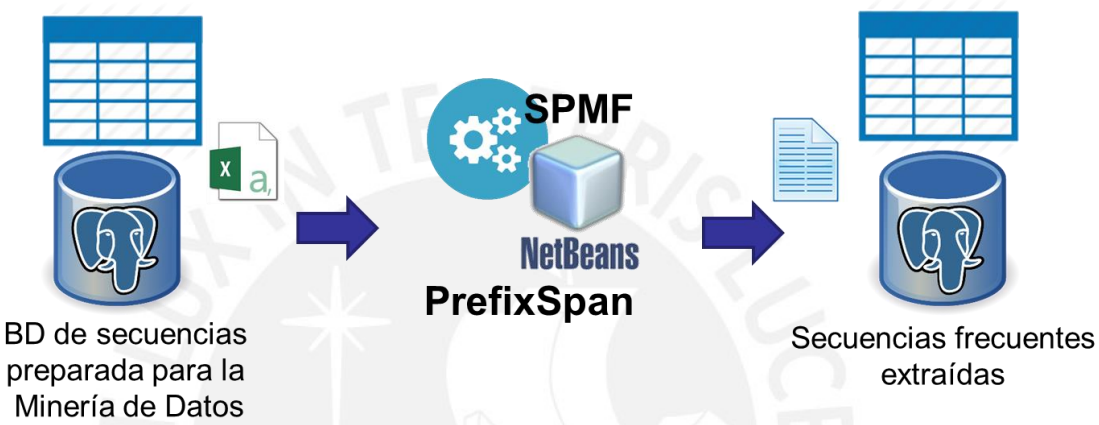

Figura 11: Proceso de utilización del algoritmo PrefixSpan. Elaboración propia.

Como ya se había mencionado previamente, en esta etapa se utilizó la librería de libre acceso SPMF para la ejecución del algoritmo PrefixSpan sobre la base de datos de secuencias generadas en la etapa de Pre-tratamiento. Como se explicó en la Sección 1.5.1, se seleccionó esta librería debido a que es de código abierto, cuenta con una gran cantidad de colaboradores que constantemente ofrecen mejoras de implementación, y actualmente se ha vuelto muy popular entre los investigadores de ciencias de la computación debido a la facilidad de poder utilizar diversos algoritmos de minería de datos sin necesidad de volverlos a implementar. Básicamente se decide utilizar PrefixSpan porque obtiene resultados de forma muy eficiente en comparación a otros algoritmos bajo un paradigma fácil de comprender (para mayor detalle revisar la Sección 2.2.2.3.2). La librería SPMF ofrece la posibilidad de usar PrefixSpan para el análisis de datos numéricos y también de cadenas de texto. Por lo tanto, se usó la segunda opción pues las secuencias que ya se habían generado en el pre-tratamiento contenían letras o símbolos de texto. Cabe mencionar que, de esta manera, el **Resultado Esperado 2.1** del presente trabajo de fin de carrera se ha realizado.

Por otro lado, los datos de entrada que se necesitaron para la ejecución del algoritmo PrefixSpan fueron dos. Primero, el minsup o soporte mínimo, el cual indica la frecuencia mínima con la que deben aparecer los patrones secuenciales del total de ríos; y segundo, el conjunto de secuencias que corresponden a los 21 ríos analizados en un archivo de texto con formato UTF-8 llamado “sequences.txt”, el cual se pudo obtener a partir de la tabla “lisofsequences”.

Para poder entender el funcionamiento del algoritmo PrefixSpan, se recuerdan las definiciones de la Sección 2.1.7.1:

- Prefijo: suponiendo que todos los ítems de están listados alfabéticamente. Dada la secuencia $\alpha = (e_1\ e_2\ \dots\ e_n)$, una secuencia $\beta = (e'_1\ e'_2\ \dots\ e'_m)$ $(m \leq n)$ es llamada un prefijo de α sí y sólo sí (1) $e'_i = e_i$ para $(i \leq m-1)$; (2) $e'_m \subset e_m$; y (3) todos los ítems en $(e_m - e'_m)$ están alfabéticamente después de aquellos en e'_m.
- Proyección: dadas las secuencias α y β, tal que β es subsecuencia de α. Una subsecuencia α’ de la secuencia α es llamada una proyección de α con respecto a β sí y sólo sí (1) α’ tiene como prefijo a β y (2) no existe una supersecuencia α’’ de α’ tal que α’’ sea una subsecuencia de α y también tenga como prefijo a β.
- Postfijo: sea $\alpha' = (e_1\ e_2\ \dots\ e_n)$ la proyección de α respecto al prefijo $\beta = (e_1\ e_2\ \dots\ e_{m-1}\ e'_m)$ $(m \leq n)$. La secuencia $\gamma = (e''_m\ e_{m+1}\ \dots\ e_n)$ es llamado el postfijo de α con respecto al prefijo β, donde $e''_m = (e_m - e'_m)$
- Base de datos proyectada: sea α un patrón secuencial en la base de datos S. La base de datos proyectada – α, denotada como $S|_\alpha$, es el conjunto de posfijos de secuencias en S con respecto al prefijo α.

El pseudocódigo del algoritmo PrefixSpan se muestra en el Algoritmo 2, en el cual, S es la base de datos de secuencias, minsup es el soporte mínimo deseado, α y α’ son patrones secuenciales, $S|_\alpha$ y $S|_{\alpha'}$ son bases de datos de secuencias proyectadas respecto a α y α’ respectivamente, y l es el tamaño de α.

Inicio PrefixSpan (S, minsup)

1. Escanear $S|_\alpha$ una vez, encontrar el conjunto de items frecuentes b tal que
   (a) b pueda ser ensamblado al final α para formar un patrón secuencial
   (b) (b) pueda ser adjuntado a α para formar un patrón secuencial
2. Para cada ítem frecuente b, adjuntarlo a α para formar un patrón secuencial α’, y salida α’

3. Para cada α', construir la base de datos proyectada - α' $S|_{\alpha'}$, y llamar a PrefixSpan (α', l+1, $S|_{\alpha'}$)

Fin PrefixSpan

Algoritmo 2: Pseudocódigo del algoritmo PrefixSpan. Extraído de (Pei, Han y otros 2001).

Luego de la ejecución del algoritmo para un minsup igual a 16, se obtuvieron los patrones secuenciales deseados en otro archivo de texto con formato UTF-8 llamado "output.txt", los cuales fueron aproximadamente tres mil, como se muestra en la Figura 12.

```
bod_<=1.7004 -1  #SUP: 17
bod_<=1.7004 -1 temp_(10.3725,11.5717] -1  #SUP: 17
bod_<=1.7004 -1 temp_(10.3725,11.5717] -1 temp_(10.3725,11.5717] -1  #SUP: 15
bod_<=1.7004 -1 temp_(10.3725,11.5717] -1 ss_<=8.8446 -1  #SUP: 15
bod_<=1.7004 -1 temp_(10.3725,11.5717] -1 bod_<=1.7004 -1  #SUP: 15
bod_<=1.7004 -1 ss_<=8.8446 -1  #SUP: 15
bod_<=1.7004 -1 ss_<=8.8446 -1 temp_(10.3725,11.5717] -1  #SUP: 15
bod_<=1.7004 -1 ss_<=8.8446 -1 ss_<=8.8446 -1  #SUP: 15
bod_<=1.7004 -1 ss_<=8.8446 -1 ss_<=8.8446 -1 temp_(10.3725,11.5717] -1  #SUP: 15
bod_<=1.7004 -1 bod_(1.7004,2.3983] -1  #SUP: 15
bod_<=1.7004 -1 bod_<=1.7004 -1  #SUP: 15
bod_<=1.7004 -1 bod_<=1.7004 -1 temp_(10.3725,11.5717] -1  #SUP: 15
bod_<=1.7004 -1 bod_<=1.7004 -1 bod_<=1.7004 -1  #SUP: 15
bod_<=1.7004 -1 bod_<=1.7004 -1 bod_<=1.7004 -1 bod_<=1.7004 -1  #SUP: 15
ph_<=7.5307 -1  #SUP: 15
```

Figura 12: Porción de secuencias frecuentes del archivo output.txt, luego de la ejecución del algoritmo Prefix-Span. Elaboración propia.

Estos resultados representan el **Resultado Esperado 2.2** del presente trabajo de fin de carrera. Así mismo, representan el comportamiento de la contaminación de los ríos en Reino Unido en base a las características y propiedades seleccionadas (temperatura, pH, restos de sólidos y la demanda biológica de oxígeno) a través de los patrones secuenciales como se muestra en la figura anterior.

En el archivo de texto, cada línea es una secuencia frecuente. Cada secuencia puede tener múltiples *itemsets* (separados por -1) y cada *itemset* puede contener múltiples *items* (separados por espacios en blanco). Al final de cada secuencia, aparece el soporte, el cual especifica el número de apariciones de dicha secuencia frecuente en el conjunto total de secuencias de los 21 ríos del Reino Unido estudiados.

3.3.3.2 **Interpretación cuantitativa del desempeño del algoritmo PrefixSpan**

A continuación, se detallan los resultados obtenidos por medio de un análisis cuantitativo del desempeño del algoritmo PrefixSpan contra el desempeño del algoritmo BIDE, el cual es otro de los algoritmos de patrones secuenciales más importantes en la actualidad. Esto corresponde a la realización del **Resultado Esperado 2.3** del presente trabajo de fin de carrera.

Se decide realizar la comparación contra el algoritmo BIDE debido al buen desempeño que posee frente otros algoritmos de extracción de patrones secuenciales, en términos de tiempo de procesamiento y memoria. Para más detalle de este algoritmo ver Sección 2.2.2.3.4 o revisar (Wang, Han 2004).

De la misma manera que PrefixSpan, se usó la librería SPMF para la ejecución del algoritmo BIDE. Este algoritmo también requiere como parámetros de entrada una base de datos de secuencias y un soporte mínimo.

Cada algoritmo se ejecutó diez veces. Los archivos utilizados fueron construidos a través de la implementación de un programa, de manera que aleatoriamente creara distintas bases de datos de secuencias. Sin embargo, con la finalidad de poder simular distintos escenarios de la base de datos de secuencias original, se consideraron aproximadamente la misma cantidad de secuencias, cantidad de ítemsets, y cantidad de ítems, así como el mismo formato requerido por SPMF en el programa (ver Anexo 3 sobre el código del programa y un ejemplo obtenido). Por otro lado, se utilizó el soporte mínimo igual a 16, debido a que se obtiene una cantidad significativa de patrones para analizar en ambos algoritmos respectivamente con este valor.

3.3.3.2.1 **Resultados**

En este caso, la función objetivo (f.o.) es maximizar la cantidad de cambios que las características (ítems) hayan sufrido en el tiempo, sobre todas las secuencias frecuentes extraídas. Se escoge esta f.o. ya que es posible conseguir información más útil e interesante sobre patrones frecuentes que involucren cambios en el tiempo, sobre todo por las correlaciones entre ítems que podrían originar dichos cambios. A partir de ello, en la Tabla 15 se muestran los resultados obtenidos por ambos algoritmos.

Muestra	Función objetivo usando PrefixSpan	Función objetivo usando BIDE
1	6239	4693
2	6601	5651
3	6031	4968
4	5491	4809
5	6277	5541
6	7908	7300
7	10671	9200
8	7488	5393
9	10052	8962
10	11878	10570

Tabla 15: Resultados de la función objetivo usando PrefixSpan y BIDE en diez muestras. Elaboración propia.

3.3.3.2.2 **Prueba Kolmogorov-Smirnov (K-S)**

Con la finalidad de demostrar que los resultados de la ejecución de los algoritmos presentan una distribución normal, se realiza la prueba Kolmogorov-Smirnov. A continuación, presentamos los detalles de la prueba K-S para cada uno de los algoritmos.

Prueba K-S para PrefixSpan

Procedimiento

Media	6424.50000	**Mínimo**	5491
Desviación Estándar	814.17	**Máximo**	7908
Varianza	662871.1	**Datos**	10

Resultados

Máxima Diferencia	0.386
Significancia de la prueba	0.05
Valor Crítico	0.410

Se ACEPTA la hipótesis nula

Datos Ordenados Xi	Valores Estandarizados Zi	Probabilidad Acumulada Sn(xi)	Probabilidad Acumulada Esperada Fi	Diferencias |Sn(xi) - Fi|
6239.00000	-0.23	0.100	0.410	0.310
6601.00000	0.22	0.200	0.586	0.386
6031.00000	-0.48	0.300	0.314	0.014
5491.00000	-1.15	0.400	0.126	0.274
6277.00000	-0.18	0.500	0.428	0.072
7908.00000	1.82	0.600	0.966	0.366
10671.00000	5.22	0.600	1.000	0.400
7488.00000	1.31	0.600	0.904	0.304
10052.00000	4.46	0.600	1.000	0.400
11878.00000	6.70	0.600	1.000	0.400

Prueba K-S para BIDE

Procedimiento

Media	5493.66667	**Mínimo**	4693
Desviación Estándar	967.06	**Máximo**	7300
Varianza	935199.0667	**Datos**	10

Resultados

Máxima Diferencia	0.369
Significancia de la prueba	0.05
Valor Crítico	0.410

Se ACEPTA la hipótesis nula

Datos Ordenados Xi	Valores Estandarizados Zi	Probabilidad Acumulada Sn(xi)	Probabilidad Acumulada Esperada Fi	Diferencias |Sn(xi) - Fi|
4693.00000	-0.83	0.100	0.204	0.104
5651.00000	0.16	0.200	0.565	0.365
4968.00000	-0.54	0.300	0.293	0.007
4809.00000	-0.71	0.400	0.239	0.161
5541.00000	0.05	0.500	0.520	0.020
7300.00000	1.87	0.600	0.969	0.369
9200.00000	3.83	0.600	1.000	0.400
5393.00000	-0.10	0.600	0.459	0.141
8962.00000	3.59	0.600	1.000	0.400
10570.00000	5.25	0.600	1.000	0.400

Las hipótesis para ambos algoritmos son:

H_0: El algoritmo PrefixSpan presenta una distribución normal.

H_1: El algoritmo PrefixSpan no presenta una distribución normal.

H_0: El algoritmo BIDE presenta una distribución normal.

H_1: El algoritmo BIDE no presenta una distribución normal.

Como se puede observar, en ambos casos el valor del estadístico (variable Máxima Diferencia) es menor al valor crítico para 10 muestras. Esto quiere decir que la solución está fuera de la región crítica. Por lo tanto, en ambos casos se acepta la hipótesis H_0, aceptando que los datos presentan una distribución normal.

3.3.3.2.3 **Prueba Fisher**

Luego de demostrar la normalidad de los datos mediante la prueba K-S, es necesario determinar si las soluciones presentan varianzas homogéneas o heterogéneas.

	Prefixspan	BIDE
Media	6424.5	5493.666667
Varianza	662871.1	935199.06667
Observaciones	10	10
Grados de libertad	9	9
F	0.70880214	
P(F<=f) una cola (derecha)	0.691799648	
Valor crítico para F (una cola)	3.178893104	
	Se ACEPTA la hipótesis nula	

Las hipótesis a evaluar son:

H_0: Las varianzas son homogéneas

H_1: Las varianzas son heterogéneas

Como se puede observar de los resultados, el valor del estadístico F obtenido de los resultados de los algoritmos es menor al valor crítico para la prueba, por lo que está fuera de la región crítica. Por lo tanto, se acepta la hipótesis H_0, aceptando que las varianzas son homogéneas.

3.3.3.2.4 **Prueba T-Student**

La prueba T-Student permitirá comparar la media de los resultados del algoritmo PrefixSpan con la media de los resultados del algoritmo BIDE. Para que la prueba sea válida, se debe que corroborar que los datos presenten una distribución normal y que posean varianzas homogéneas. Esto ha sido demostrado por medio de la prueba K-S y la prueba de Fisher respectivamente.

Esta prueba está dividida en dos partes. En la primera parte, se determinará si la media de los resultados obtenidos con el algoritmo PrefixSpan es igual o significativamente diferente a la media de los resultados obtenidos con el algoritmo BIDE. Si se llega a la conclusión de que las medias son diferentes, en la segunda parte se determinará cuál de las dos medias es mayor a la otra.

	Prefixspan	BIDE
Media	6424.50000	5493.66667
Varianza	662871.10000	935199.06667
Observaciones	10	10
Varianza agrupada	799035.08333	
Diferencia de medias	930.83333	
Grados de libertad	18	
Estadístico t	5.20666	
Valor crítico de t (una cola)	1.73410	
Valor crítico de t (dos colas)	2.10090	
Hipótesis medias diferentes	**Se RECHAZA la hipótesis nula**	
Hipótesis media BIDE > media PrefixSpan	**Se RECHAZA la hipótesis nula**	

Para la primera parte, se recurre a la prueba de dos colas usando como hipótesis:

H_0: La media del algoritmo PrefixSpan es igual que la media del algoritmo BIDE

H_1: La media del algoritmo PrefixSpan es diferente que la media del algoritmo BIDE

Para ello, se usa el valor crítico de Z (dos colas). Para aceptar como cierta la hipótesis nula, el valor de Z debe estar entre -2.1009 y +2.1009. Sin embargo, se observa que el estadístico es 5.20666 por lo que caería en una región crítica

haciendo que se rechace la hipótesis nula. Por lo tanto, se llega a la conclusión que tanto las medias de ambos algoritmos son diferentes.

Para la segunda parte, al haber demostrado que las medias son significativamente diferentes, se recurre a la prueba de una cola utilizando como hipótesis:

H_0: La media del algoritmo BIDE es mayor que la media del algoritmo PrefixSpan

H_1: La media del algoritmo BIDE es menor que la media del algoritmo PrefixSpan

Para ello se usa el valor crítico de la tabla Z (una cola). Para aceptar como cierta la hipótesis nula, el valor de Z debe ser mayor a -1.7341. Sin embargo, se observa que dicho valor es -5.20666 por lo que caería en una región crítica haciendo que se rechace la hipótesis nula. Se llega a la conclusión que la media del algoritmo BIDE es menor a la media del algoritmo PrefixSpan.

3.3.3.2.5 **Conclusión**

Luego de haber realizado la interpretación cuantitativa del desempeño del algoritmo PrefixSpan, comparándolo con el algoritmo BIDE, se ha podido observar que la media de la función objetivo de las soluciones del algoritmo PrefixSpan es significativamente mayor a la media de la función objetivo de las soluciones del algoritmo BIDE. Se puede concluir que el algoritmo PrefixSpan tiene mejor desempeño que el BIDE para los conjuntos de datos que se simularon a partir de la base de datos de secuencias original.

3.3.3.3 **Algoritmo de descubrimiento de los patrones que mejor describen al fenómeno estudiado**

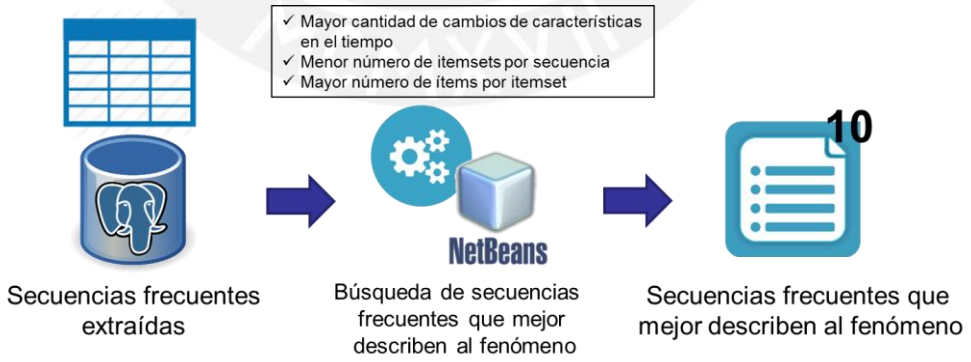

Figura 13: Proceso de búsqueda de las diez secuencias que mejor describen al fenómeno, en base a tres criterios propios. Elaboración propia.

En esta sección, se realiza el proceso mostrado en la Figura 13, donde se tuvo que elaborar un algoritmo en Netbeans en Java JDK 8 que permita obtener las diez secuencias frecuentes deseadas (ver Anexo 4). Además, se realiza la interpretación cualitativa de estas diez secuencias frecuentes con la finalidad de entender el significado de dichas secuencias frecuentes en base al contexto del fenómeno de contaminación de los ríos en Reino Unido (ver Sección 2.1.9). Todo esto corresponde a la realización del **Resultado Esperado 3.1** del presente trabajo de fin de carrera.

Con respecto al algoritmo para la obtención de las secuencias frecuentes que mejor describen al fenómeno, los criterios utilizados son la mayor cantidad (por secuencia) de cambios de las características en el tiempo, el menor tamaño de la secuencia frecuente (ver Sección 2.1.7.1) y la mayor cantidad de *items* dentro de cada *itemset* de la secuencia, de manera que se priorizarán las secuencias frecuentes con mayor cantidad de cambios de características en el tiempo, luego continuará el tamaño y finalmente la cantidad de *items* por *itemset*. El pseudocódigo se muestra en el Algoritmo 3.

Inicio findBest10FrecuentSequences(secuencias)

//Estas constantes dependen de la cantidad de secuencias que se deseen filtrar por cada criterio. Los valores escogidos en esta oportunidad podrían ser cambiados según la necesidad. Al final sólo se seleccionan 10 secuencias.

1. Inicializar constantes nSecuenciasMostChangedItems = 100, nSecuenciasNumberItems = 50, nSecuenciasNumberItemsets = 10
2. mejoresSecuencias = getBestSequencesBasedOnMostChangedItems(secuencias, nSecuenciasMostChangedItems)
3. mejoresSecuencias = getBestSequencesBasedOnNumberItems (mejoresSequencias, nSecuenciasNumberItems)
4. mejoresSecuencias = getBestSequencesBasedOnNumberItemsets (mejoresSequencias, nSecuenciasNumberItemsets)
5. Retornar mejoresSecuencias

Fin findBest10FrecuentSequences

Inicio submódulo getBestSequencesBasedOnMostChangedItems (secuencias, nSecuencias)

1. características = obtenerCaracteristicas (secuencias)

2. Para (cada secuencia seq en secuencias)
    a. Para (cada característica car en características)
        i. Flag = 1
        ii. Para (cada itemset iset en seq.itemsets)
            1. Para (cada ítem en iset.items)
                a. Si (ítem.caracteristica == car)
                    i. Si (flag ==1)
                        1. car.anteriorItem = ítem.caracteristica
                        2. Flag = 0
                    ii. Si (car.anteriorItem != ítem.caracteristica)
                        1. Car.{anteriorItem = ítem.caracteristica
                        2. Car.cambios = car.cambios + 1
                    iii. Break
    b. Sumar los cambios de todas las características y asignarlo a seq.cambios
    c. Inicializar car.cambios = 0 en características
3. Ordenar descendentemente secuencias por seq.cambios
4. Retornar las primeras nSecuencias de bdsecuencias

Fin getBestSequencesBasedOnMostChangedItems

Inicio submódulo getBestSequencesBasedOnNumberItems (secuencias, nSecuencias)

1. Ordenar descendentemente secuencias por número de items
2. Retornar las primeras nSecuencias de secuencias

Fin getBestSequencesBasedOnNumberItems

Inicio submódulo getBestSequencesBasedOnNumberItemsets (secuencias, nSecuencias)

1. Ordenar descendentemente secuencias por número de itemsets
2. Retornar las primeras nSecuencias de secuencias

Fin getBestSequencesBasedOnNumberItemsets

Algoritmo 3: Búsqueda de secuencias frecuentes que mejor describen al fenómeno. Elaboración propia.

El algoritmo toma como entrada el archivo de texto “output.txt” generado en la Sección 3.3.3.1. En base a este archivo, cada secuencia es recuperada en una lista de secuencias compuesta por una lista de *itemsets*, los cuales están compuestos

de una lista de ítems (ver las clases secuencia, itemset e ítem del Anexo 6). Por ejemplo, si se tiene la siguiente secuencia frecuente:

temp_(10.3582,11.5801] ph_>7.8469 -1 ph_>7.8469 -1 #SUP: 16

Entonces, cada *itemset* será el conjunto de propiedades físico-químicas separadas por "-1". En este caso, se tienen dos *itemsets* donde el primero contiene dos *items* y el segundo, un *item*.

Luego de haber cargado la lista de secuencias frecuentes, ésta es ordenada descendentemente según la cantidad de cambios que las características hayan tenido en el tiempo, de manera que, se muestran al inicio las secuencias con mayor cantidad de cambios (submódulo *getBestSequencesBasedOnMostChangedItems* del Algoritmo 3). Después de extraer las primeras *nSecuenciasMostChangedItems* secuencias, estas se ordenan descendentemente según la mayor cantidad de ítems por itemset, debido a que esto permitirá analizar posibles asociaciones entre las propiedades físico-químicas que cambian en el tiempo (submódulo *getBestSequencesBasedOnNumberItems* del Algoritmo 3). Luego de extraer las primeras *nSecuenciasNumberItemsets* secuencias, estas se ordenan ascendentemente según la cantidad de itemsets o tamaño de secuencia, de manera que las secuencias frecuentes más cortas queden al inicio de la lista. (submódulo *getBestSequencesBasedOnNumberItemsets* del Algoritmo 3). Finalmente, se seleccionan las *nSecuenciasNumberItems* primeras secuencias frecuentes que son diez; estas serán consideradas como las secuencias frecuentes que mejor describen al fenómeno de contaminación de los ríos en Reino Unido. Los resultados se imprimieron en el archivo "best_sequences.txt".

Con respecto a la interpretación cualitativa, se explica como ejemplo el significado de la siguiente secuencia frecuente extraída del archivo "output.txt":

temp_(10.3725,11.5717] -1 temp_(10.3725,11.5717] ss_>17.3205 -1 temp_>11.5717

Entonces, se puede observar que esta secuencia frecuente está compuesta de tres *itemsets* o conjunto de eventos temporales. Al inicio la temperatura tenía un valor medio. Luego de un tiempo, bajó esta temperatura intermedia y se registró la presencia de altos niveles de restos sólidos. Después, la temperatura se incrementó, pudiendo haber superado el límite de riesgo según los estándares de temperatura de la Sección 2.1.9. Este hecho pudo haber afectado el crecimiento y desarrollo de las especies acuáticas, el éxito en reproducción, y la sobrevivencia de

estas especies, como se explicó en la Sección 2.1.9. Esta secuencia tiene como soporte 16, es decir que aparece en 16 ríos de los 21 estudiados (más del 50%).

3.4 Conclusión

Luego de todo lo anteriormente explicado, se concluye haber culminado con el objetivo de este capítulo, el cual era especificar el fenómeno a estudiar y aplicar la metodología KDD para extraer patrones secuenciales que describan su comportamiento. Para ello, se utilizaron diversas herramientas informáticas que permitan automatizar ciertas actividades en el desarrollo del presente trabajo de fin de carrera. Además, se pudo culminar con los **Objetivos Específicos 1, 2 y 3** del presente trabajo de fin de carrera luego de haber finalizado con el desarrollo de los resultados esperados respectivos.

CAPÍTULO 4: PROTOTIPO DE VISUALIZACIÓN

En este capítulo, se explica la construcción de una herramienta de visualización que permita representar las secuencias frecuentes obtenidas en el capítulo previo en un mapa con los ríos estudiados. Primero, se describen las actividades necesarias para poder obtener los ríos donde ocurren las diez secuencias que mejor describen el fenómeno estudiado. Segundo, se explican los pasos para poder obtener los puntos geográficos de los ríos por cada una de las diez secuencias frecuentes. Finalmente, se detalla el proceso de construcción de la herramienta de visualización, la cual utiliza los puntos geográficos hallados previamente para mostrarlos en un mapa según la secuencia frecuente seleccionada.

4.1 Objetivo del prototipo de visualización

El objetivo de este programa de visualización es representar en un mapa las secuencias frecuentes que mejor describen al fenómeno seleccionado (la contaminación de los ríos en Reino Unido), mostrando las características que representaron cambios temporales y que podrían ayudar a los expertos a identificar la posible polución de los ríos. Así mismo, este capítulo corresponde al desarrollo y cumplimiento del Objetivo Específico 4 del presente trabajo de fin de carrera.

4.2 Obtención de los ríos donde ocurren las secuencias frecuentes que describen mejor al fenómeno estudiado

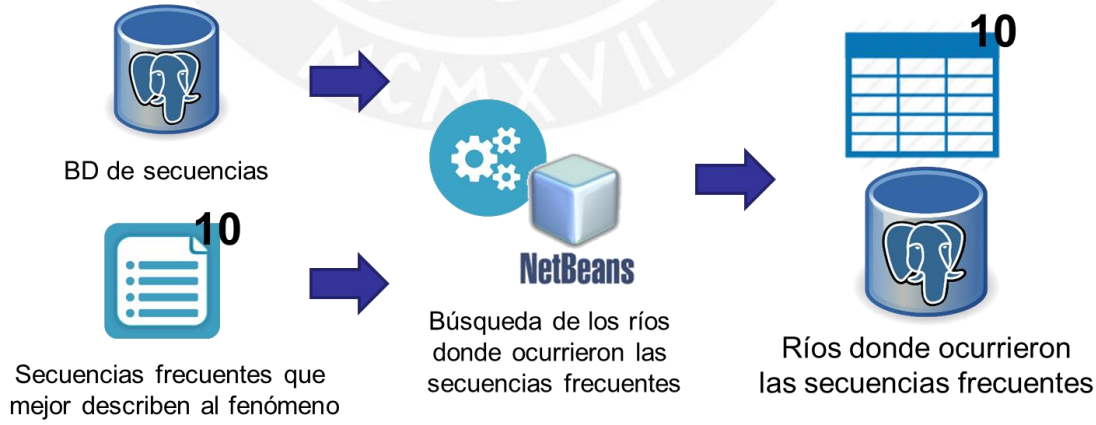

Figura 14: Proceso de búsqueda de ríos donde ocurrieron las diez secuencias que mejor describen al fenómeno. Elaboración propia.

Debido a que la herramienta de visualización muestra en un mapa los ríos donde ocurrieron las secuencias frecuentes que mejor describen al fenómeno estudiado, es necesario implementar un algoritmo que permita realizar la búsqueda de los ríos donde ocurrió cada secuencia frecuente. Para ello, se ha usado Netbeans como IDE para la construcción del programa en Java JDK 8 (ver Anexo 5). El pseudocódigo propuesto se muestra en el Algoritmo 4.

Inicio findRiversOfBestSequences (mejoresSecuencias, bdSecuencias)

1. Para (cada mejor secuencia msec en mejoresSecuencias)
    a. Para (cada secuencia dbsec en bdSecuencias)
        i. Si (mejorSecuenciaEncontrada(msec, dbsec)
            1. Guardar dbsec.id, dbsec.nombreRio, dbsec.latitud y dbsec.longitud en msec

Fin findRiversOfBestSequences

Inicio submódulo mejorSecuenciaEncontrada (msec, dbsec)

1. Inicializar sequenceFound = 0, itemsetFound = 0 y lastPostFound = 0
2. Para (cada itemset miset de msec)
    a. Para (cada i desde lastposfound, tal que i < tamaño de dbsec.itemsets)
        i. Si (itemsetFound = itemsetEncontrado (miset, dbsec.itemsets.get(i)))
            1. Si miset == ultimo itemset en msec
                //Si el último itemset ha sido hallado, entonces la secuencia ha sido hallada
                a. sequenceFound = 1
            2. Sino
                a. lastPosFound = i + 1
            3. break

    //Si ningún itemset ha sido hallado, entonces la secuencia no será encontrada. Retornar 0

    b. Si (!itemsetFound)
        i. Break

    //Si la secuencia ha sido hallada, no iterar más y retornar 1

    c. Si (sequenceFound)
        i. break
3. Retornar sequenceFound

Fin mejorSecuenciaEncontrada

Inicio submódulo itemsetEncontrado (miset, dbiset)

1. Inicializar itemsetFound = 0, itemFound = 0j
2. Para (cada ítem mitem en miset.items)
    a. Para (cada ítem dbitem en dbiset.items)
        i. Si (ítemFound = (mitem == dbitem))
            1. Si (mitem == ultimo ítem en miset)
                //Si el último item ha sido hallado, entonces el itemset ha sido hallado
                a. itemsetFound = 1
            2. break

    //Si ningún ítem ha sido hallado, entonces el itemset no será encontrado. Retornar 0

    b. Si (!itemFound)
        i. Break

    //Si el ítemset ha sido hallado, no iterar más y retornar 1

    c. Si (itemsetFound)
        i. Break
3. Retornar itemsetFound

Fin itemsetEncontrado

Algoritmo 4: Búsqueda de los ríos donde ocurrieron las secuencias frecuentes. Elaboración propia.

Este algoritmo toma como datos de entrada la tabla “listofsequences” con las secuencias de los 21 ríos del Reino Unido (ver Tabla 14) y el archivo “best_sequences.txt” con las secuencias frecuentes que mejor describen al fenómeno seleccionado (ver Sección 3.3.3.3). Este algoritmo compara a nivel de secuencia (submódulo mejorSecuenciaEncontrada del Algoritmo 4), itemset e item (submódulo mejorItemsetEncontrado del Algoritmo 4) ambas fuentes de datos, de manera que cada secuencia frecuente del archivo “output.txt” se busque en cada secuencia de la tabla “listofsequences”, ya que cada secuencia de la tabla le corresponde a un río del Reino Unido. Al final, lo que se obtiene es la creación de las diez tablas con los ríos donde ocurrieron las diez secuencias frecuentes que mejor describen al fenómeno, de manera que ya contienen la columna the_geom completa.

Por ejemplo, dada la secuencia frecuente “SF” del archivo “output.txt”:

SF = temp_(10.3582,11.5801] ss_>17.3205 **-1** ph_>7.8469 **-1 -2** #sup: 16

Y dada la secuencia “S” del río “A” de la tabla “listofsequences”:

S = temp_(10.3582,11.5801] ph_>7.8469 ss_>17.3205 bod_>22.1594 **-1**

temp_>11.5801 ph_>7.8469 ss_(8.8446,17.3205] bod_>22.1594 **-1**

temp_>11.5801 ph_>7.8469 ss_(8.8446,17.3205] bod_>22.1594 **-1**

…. **-1 -2**

Entonces, SF está incluida en S, debido a que cada ítemset de la secuencia SF está incluido **respetando el orden temporal** en algún itemset de la secuencia S. En este caso, el itemset temp_(10.3582,11.5801] ss_>17.3205 aparece antes que el ítemset ph_>7.8469. Por lo tanto, el río A formaría a ser parte de la lista donde la secuencia frecuente ocurre.

Es necesario notar que dentro de un itemset las propiedades físico-químicas del fenómeno no necesariamente deben estar ordenadas ni deben ser consecutivas, como fue el caso del itemset temp_(10.3582,11.5801] ss_>17.3205. Por otro lado, para que la secuencia frecuente esté incluida en alguna secuencia se necesita que todos sus *itemsets* sean encontrados en el orden que aparecen y que no necesariamente sean consecutivos, como ocurrió con la secuencia frecuente SF.

La estructura de cada una de las diez tablas generadas se muestra en la Tabla 16. Estas tablas serán utilizadas en la visualización de los ríos posteriormente.

Nombre de la Columna	Tipo de dato	¿Es nulo?	Descripción
ID	Int	No	Llave primaria
River_name	Text	No	Región geográfica y nombre del río donde se tomaron los datos
Latitud	Double precision	No	Coordenada espacial latitud
Longitud	Double precision	No	Coordenada espacial longitud
The_geom	Geometry	No	Punto geométrico en base a la latitud, longitud, y proyección espacial EPGS: 4326

Tabla 16: Estructura de cada una de las diez tablas que contienen los ríos donde ocurrieron las secuencias que mejor describen al fenómeno. Elaboración propia.

4.3 Obtención de los puntos geográficos

Figura 15: Proceso de creación y carga de archivos KML. Elaboración propia.

Con la finalidad de mostrar los ríos donde ocurren las secuencias frecuentes que mejor describen al fenómeno estudiado, es necesario mostrar los puntos geográficos que representen las estaciones donde se tomaron los datos del río en un mapa. Para ello, se ha utilizado QGis para la visualización de la proyección espacial de los puntos geográficos y Google Maps para la carga, edición y presentación de las capas espaciales.

Primero, se utilizó la funcionalidad de añadir capa PostGis de QGis para recuperar las diez tablas generadas en la Sección 3.3.3.3 de las secuencias frecuentes que mejor describen al fenómeno seleccionado. De esta manera, se pudo revisar que las proyecciones hayan sido creadas correctamente. Además, se pudo conseguir una base de datos de visualización de los ríos en Gran Bretaña (OSOpenData, 2015) que cubren a los países de Inglaterra, Gales y Escocia. Esta base de datos permitió mostrar la compleja red de ríos existente en el Reino Unido. Así mismo, QGis permite instalar un complemento llamado “Open Layers plugin” que contiene las opciones de visualización de distintos mapas como Google Maps, Bing Maps, Apple Maps, entre otros. De esta forma, variando entre las tres capas definidas se puede obtener lo que se muestra en la Figura 16. Se muestran las 21 estaciones de los 21 ríos donde se tomaron los registros de sus propiedades físico-químicas.

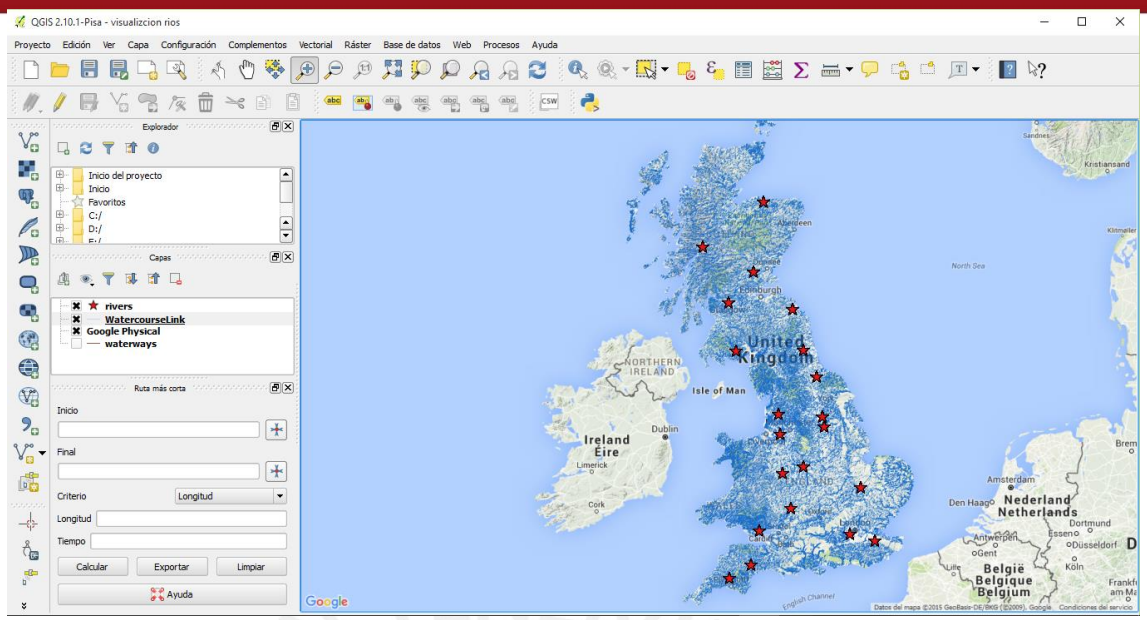

Figura 16: Ejemplo de visualización de las estaciones de los 21 ríos analizados sobre toda la red de ríos de Gran Bretaña visto desde Google Maps. Elaboración propia.

Segundo, para la creación de los archivos KML y su posterior implementación sobre el mapa de Google, se usó la funcionalidad de exportación al formato KML de QGis. De esta manera, se guardaron los diez archivos KML de las diez tablas generadas en la Sección 3.3.3.3, también se guardó un archivo KML de la tabla "listofsequences", donde se muestran todas las 21 estaciones, así mismo se guardaron dos archivos KML de la base de datos de ríos de Inglaterra, Galés y Escocia luego de haber filtrado sólo los 21 ríos seleccionados (se dividió en dos debido a que Google no admite archivos KML mayores a 3 MB).

Finalmente, todos los archivos fueron subidos a Google Maps. Se tuvo que brindar permisos para todo el público con la finalidad de poder conseguir el script de inserción en sitios web. Por lo tanto, se tuvo que asignar a cada una de las diez mejores secuencias frecuentes el URL del script (ver Anexo 7).

4.4 Construcción del prototipo de visualización

En esta sección, se describen las actividades de la construcción del programa de visualización para la representación de las diez secuencias frecuentes que mejor describen al fenómeno estudiado, mostrando en un mapa los ríos donde ocurrió cada secuencia frecuente. Para esto, se hizo uso de JQuery 1.11.3 como herramienta para el desarrollo web, el API de Google Maps para la visualización de los ríos y estaciones en el mapa, y la base de datos de secuencias frecuentes.

Primero, se planteó el diseño del programa con la finalidad de tener una visión más clara de cómo se realizaría la funcionalidad de búsqueda de las secuencias frecuentes. Con respecto a la interfaz gráfica, se colocó un título, una breve descripción de este trabajo de fin de carrera, un Google Maps en la parte central izquierda, y una tabla con el conjunto de las secuencias frecuentes que mejor describen al fenómeno en la parte central derecha.

Por otro lado, con respecto a la arquitectura, tanto el servidor web como la base de datos (archivo JSON con las secuencias que mejor describen al fenómeno) se trabajaron localmente. Cuando se envía la petición de búsqueda de alguna secuencia frecuente, se manda una solicitud de búsqueda al archivo JSON y se obtiene la ruta del archivo KML asociado a la secuencia frecuente buscada, la cual es retornada para que el mapa se refresque y vuelva a cargar.

Segundo, se realizó la implementación del diseño en un programa web. Se hizo uso de JQuery 1.11.3 y del API de Google Maps para poder realizar peticiones asíncronas a la base de datos y así obtener las rutas de los archivos KML, de manera que luego estos sean superpuestos sobre el mapa de Google según la selección de la secuencia frecuente que haya realizado el usuario. Finalmente, la culminación de este desarrollo corresponde a la finalización del **Resultado Esperado 4.1** del presente trabajo de fin de carrera. La Figura 17 muestra el programa web funcionando luego de su implementación.

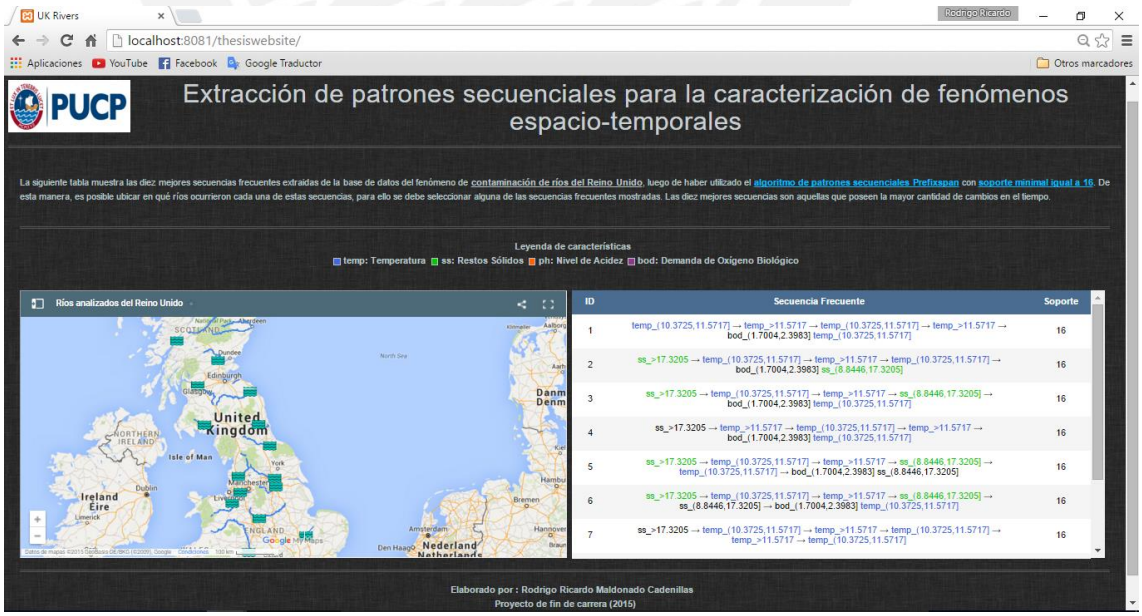

Figura 17: Prototipo de visualización. Elaboración propia.

4.5 Conclusión

Luego de todo lo explicado, se concluye haber culminado con el objetivo de este capítulo, el cual era construir un programa de visualización que represente las secuencias frecuentes que mejor describen al fenómeno de la contaminación de los ríos en Reino Unido. Además, se concluye haber finalizado el **Objetivo Específico 4.** del presente trabajo de fin de carrera, tras haber terminado con el **Resultados Esperado 4.1** dentro del presente capítulo.

CAPÍTULO 5: CONCLUSIONES

El presente trabajo de fin de carrera brinda una alternativa de solución consistente para el problema de la mejora del entendimiento de los fenómenos complejos, con la finalidad de entender su comportamiento y así poder prever acontecimientos que se deriven de ellos. Por lo tanto, se propuso un proceso en base a la metodología KDD que pudiera hacer uso de técnicas de minería de datos entre otras herramientas para poder cumplir con dicho propósito.

Sin embargo, antes de comenzar con el desarrollo del presente trabajo de fin de carrera, era necesario definir y describir el fenómeno a estudiar. Se eligió a la contaminación de los ríos en Reino Unido debido a la complejidad de estudio de este fenómeno y a la necesidad existente de esta región por mejorar la calidad de sus fuentes de agua, como sus ríos. Es por ello que se realizó una investigación profunda sobre este fenómeno y se pudo conseguir bastante información relevante para el análisis, así como una base de datos con las características físico-químicas de estos ríos.

De esta manera, se definieron objetivos específicos que permitan medir el cumplimiento del presente trabajo de fin de carrera.

Con respecto al objetivo específico de preparar y transformar los datos recolectados para mejorar su calidad y facilitar el manejo de los mismos, se tuvieron que realizar distintas actividades que permitieran cumplir con dicho fin. Primero, se empleó la herramienta Orange Biolab para la limpieza de los datos faltantes, de manera que estos puedan ser completados en base al método de promedios o método frecuentista. Segundo, se empleó una técnica de discretización de los datos en base a al método de frecuencias equivalentes con la ayuda de Orange Biolab. Finalmente, se utilizó una función en PostgreSQL para la preparación de los datos para el algoritmo PrefixSpan.

Con respecto al objetivo específico de extraer patrones secuenciales aplicando el algoritmo de Minería de Datos PrefixSpan, se utilizó la librería SPMF de Philippe Fournier-Viger para hacer uso de la implementación del algoritmo PrefixSpan sobre los datos preparados en la etapa de pre-tratamiento de datos. Así mismo, se realizó una interpretación cuantitativa del desempeño del algoritmo PrefixSpan comparándolo contra el algoritmo BIDE, aplicando diversos métodos estadísticos. Al final, se pudo comprobar que PrefixSpan fue el algoritmo más adecuado para las muestras simuladas a partir de la base de datos de secuencias original

Con respecto al objetivo específico de diseñar un algoritmo para descubrir los patrones más relevantes y que mejor describen al fenómeno, se consideraron tres criterios 1) secuencias con mayor cantidad de cambios en el tiempo, 2) secuencias con la menor longitud, y 3) secuencias con la mayor cantidad de ítems. A partir de ello, se pudieron obtener sólo diez secuencias, las cuales serían consideradas las mejores. Además, se realizó un análisis cualitativo de la semántica de los patrones, de manera que, en base a la investigación realizada sobre el fenómeno seleccionado, se realizó la interpretación de los resultados.

Con respecto al objetivo específico de implementar un programa de visualización de los resultados obtenidos, se utilizaron diversas herramientas para lograr la implementación del programa de visualización, tales como PostGis, QGIS, GeoServer y JQuery.

Finalmente, el cumplimiento de los objetivos previamente mencionados ha conllevado a la culminación exitosa de este trabajo de fin de carrera. Se espera que con este aporte se promueva a la investigación en este tipo de temas dentro del campo de Minería de Patrones Secuenciales.

REFERENCIAS BIBLIOGRÁFICAS

AGRAWAL R., SRIKANT R., 1995. Mining Sequential Patterns. Proceedings of the Eleventh International Conference on Data Engineering, IEEE CS: 3-14

ALIOTTA M.,2012. Data mining techniques on volcano monitoring. Tesis para obtener el Grado de Doctorado en Informática. Catania : Catania.

ANDERSCH Ch., F., 2006. Graphical Presentation of Sequential Patterns. Tesis para obtener el Grado de Informática de Negocios. Böblingen : Wismar.

ANTUNES C., OLIVEIRA A. 2004. Sequential pattern mining algorithms: Trade-offs between speed and memory. In Proceedings of the Workshop on Mining Graphs, Trees and Sequences.

ARORA A., MALHOTRA P.K., MARWAH S., BHARDWAJ A., DAHIYA S., 2009. Data Mining Techniques and Tools for Knowledge Discovery in Agricutural Datasets. Winter School.

AYRES J., FLANNICK J., GEHRKE J., AND YIU T., 2002. Sequential Pattern Mining Using a Bitmap Representation. Proceedings of Conference on Knowledge Discovery and Data Mining. pp. 429–435

BHENSDADIA C., KOSTA Y., 2012. An Efficient Algorithm for Mining Frequent Sequential Patterns and Emerging Patterns with Various Constraints. IJSCE pp. 59-65

CHENG H., YAN X., HAN J., 2004. IncSpan: incremental mining of sequential patterns in large database. SIGKDD pp 527–532

DATA.GOV.UK OPEN UP GOVERNMENT, 2013. What's data.gov.uk all about?. Data.gov.uk Open Up

DEMŠAR J., CURK T. Y ERJAVEC A., 2013. Orange: Data Mining Toolbox in Python. Orange Biolab 5.0.1. http://orange.biolab.si/. Fecha de consulta: 30/08/2015

FAYYAD U., PIATETSKY-SHAPIRO G., SMYTH P., 1996a. Knowledge Discovery and Data Mining: Towards a Unifying Framework. AAI/MIT press. pp. 82-88

FAYYAD U., PIATETSKY-SHAPIRO G., SMYTH P., 1996b. From Data Mining to Knowledge Discovery in Databases. AAAI. pp. 37-54

FAYYAD U., PIATETSKY-SHAPIRO G., SMYTH P., 1996c. Advances in knowledge discovery and data mining. Primera Edición. AAAI Press.

FAYYAD U., PIATETSKY-SHAPIRO G., SMYTH P., 2013. The KDD Process for Extracting Useful Knowledge from Volumes of Data. Communications of the ACM. Volume 39 Issue 11, Nov. 1996. Pages 27-34

FAYYAD U., STOLORZ P., 1997. Data mining and KDD: Promise and challenges. FGCS, 13: 99-115

FONT, Yadira 2013. Análisis comparativo de algoritmos utilizados en la minería de secuencias frecuentes. Para optar la Maestría en Cibernética Aplicada, mención Minería de Datos. La Habana: ICIMAF.

FOURNIER-VIGER P., GOMARIZ GUENICHE, T. A., SOLTANI A., WU. C., TSENG V. S., 2014. SPMF: a Java Open-Source Pattern Mining Library. http://www.philippe-fournier-viger.com/spmf/index.php. Fecha de consulta: 30/08/2015

GAROFALAKIS M., RASTOGI R., SHIM, K., 1999. SPIRIT: Sequential pattern mining with regular expression constraints. International Conference on Very Large Databases. pp 223–234.

GEOSERVER, 2015a. Publishing a PostGis Table. Geoserver 2.9.x. http://docs.geoserver.org/latest/en/user/gettingstarted/postgis-quickstart/index.html. Fecha consulta: 04/10/2015.

GEOSERVER, 2015b. GeoServer. GeoServer. http://geoserver.org/. Fecha consulta: 04/10/2015.

GOOGLE DEVELOPERS, 2015. Google Maps Javascript Api, Map Types. Google Developers.
https://developers.google.com/maps/documentation/javascript/maptypes#PixelCoordinates. Fecha de consulta: 30/08/2015

GOUDA K., HASSAAN M., ZAKI M., 2009. Prism: An effective approach for frequent sequence mining via prime-block encoding. Journal of Computer and System Sciences, pp. 1-15

Government. http://data.gov.uk/data/search. Fecha de consulta: 30/08/2015

GOZZARD EMMA, 2014. The River Thames Initiative. Centre for Ecology & Hydrology. hhttp://www.ceh.ac.uk/our-science/projects/river-thames-initiative. Fecha de consulta: 26/10/2015

HALAWANI S., KHAN M., 2010. A Study of Customer Behavior and Sales Promotion Using Generalized Sequential Pattern Mining. In International Journal of Engineering and Technology pp 149-154

HAN J., CHENG H., XIN D., YAN X., 2006. Frequent pattern mining: current status and future directions. DMKD. Vol 15, Num 1, pp 55-86.

HAN J., PEI J., MORTAZAVI-ASL B., CHEN Q., DAYAL U., HSU M., 2000. Freespan: frequent pattern-projected sequential pattern mining. SIGKDD pp 355–359.

HUANG T., 2012. Mining the change of customer behavior in fuzzy time-interval sequential patterns. Applied Soft Computing pp 1068–1086

IZADI M., BUCKERIDGE D., CHARLAND K., 2011. The first international conference on advances in Information Mining and Management. IMMM. pp 1-6

JAHANIAN K., 2011. Using Sequential Pattern Mining in Protein Sequences Discovery with Gap. In Australian Journal of Basic and Applied Sciences pp 1476-1480

KIM M., SHIN H., CHUNG T., JOUNG J., KIM J., 2011. Extracting regulatory modules from gene expression data by sequential pattern mining. In BMC Genomics

KRZYSZTOF C., WITOLD P., ROMAN S., LUKASZ K., 2007 Data Mining – A Knowledge Discovery Approach.

KUM H., PEI J., WANG W., DUNCAN D.,2002. ApproxMAP: Approximate mining of consensus sequential patterns. In Mining Sequential Patterns from Large Data Sets. pp 138 – 160

LEDENEVA Y., 2008. Effect of Preprocessing on Extractive Summarization with Maximal Frequent Sequences. In Springer-Verlag Berlin Heidelberg, pp. 123–132

LIU H., MOTODA H.,1998. Feature Selection for Knowledge Discovery and Data Mining. Springer Science & Business Media.

MADALENA M., YAMAGUCHI K., 2012 Advantages in Data Mining Knowledge Discovery and Applications. INTECH.

MAIMON O., ROKACH L.,2005. Decomposition methodology for knowledge discovery and data mining: Theory and Applications. Primera Edición ed. Singapore: World Scientific.

MARUTHAMUTHU R., KUMAR A.,2012. Grouped Frequent Sequential Pattern Analysis for Satellite Image Time Series (SITS) Data in Image Mining. NCACSA.

MASSEGLIA, F., CATHALA, F., AND PONCELET, P. 1998. The PSP approach for mining sequential patterns. European Symposium on Principles of Data Mining and Knowledge Discovery. pp 176–184.

MDN - MOZILLA DEVELOPER NETWORK, 2015. About jQuery. Mozilla Developer Net-work. http://api.jquery.com/. Fecha de consulta: 30/08/2015

MEISS M., DUNCAN J., GONÇALVES B., RAMASCO J., MENCZER F., 2009. What's in a session: tracking individual behavior on the web. In Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, pp. 173–182.

MOONEY C., RODDICK J., 2013. Sequential Pattern Mining – Approaches and Algorithms. ACM Comp.Surv. 1-46

NIZAR R., EZEIFE C., 2010. A taxonomy of sequential pattern mining algorithms. ACM Comp. Surv.

OSOPENDATA, 2015. OS Open Rivers. OSOpenData. https://www.ordnancesurvey.co.uk/opendatadownload/products.html;jsessionid=FFEE86D4FD3E0FFC5E444AFAAE948CAF. Fecha consulta: 04/10/2015.

PEI J., HAN J., 2000. Mining Frequent Patterns by Pattern – Growth: Methodology and Implications. SIGKDD. Pp 14-20

PEI J., HAN J., MORTAZAVI-ASL B., PINTO H., CHEN Q., DAYAL U., HSU M., 2001. PrefixSpan mining sequential patterns efficiently by prefix projected pattern growth. ICDE. pp 215–226.

PIATETSKY-SHAPIRO, G., 1991. Knowledge Discovery in Real Databases, AI Magazine, Winter 1991

POSTGIS, 2015. Postgis features. POSTGIS. http://postgis.net/features. Fecha consulta: 04/10/2015.

POSTGRESQL-ES PORTAL EN ESPAÑOL SOBRE POSTGRESQL, 2010. Sobre PostgreSQL. PostgreSQL-es. http://www.postgresql.org.es/. Fecha de consulta: 30/08/2015

QGIS, 2015. Guía de usuario de QGIS. QGIS. http://docs.qgis.org/2.8/es/docs/user_manual/preamble/features.html. Fecha consulta: 04/10/2015.

RAMESHI C., CHALAPATI K., GOVERDHAN A., 2011. A semantically enriched web usage based recommendation model. IJCSIT

SHETTAR R., 2012. Sequential Pattern Discovery from Web Log Data. In International Journal of Computer Applications.

SILVA E., MENEZES E.,2005. Metodologia da Pesquisa e Elaboração de Dissertação. Florianópolis: Universidade Federal de Santa Catarina - UFSC.

SLIMANI T., LAZZEZ A., 2013. Sequential Mining: Patterns and algorithms analysis. IJCER, Vol. 2, Num. 5, pp 639-647

SRIKANT R., AGRAWAL R., 1996. Mining sequential patterns: Generalizations and performance improvements. Int. Conference on Extending Database Technology. pp 3–17.

STOLORZ P., DEAN Ch.,1996. Quakefinder: A scalable data mining system for detecting earthquakes from space. AAAI. pp 208-213

TRYFONA N. Modeling Phenomena in Spatiotemporal Databases: Desiderata and Solutions. 9th International Conference, DEXA'98, Vienna, Austria.

TUMASONIS R., DZEMYDA, G., 2004. The probabilistic algorithm for mining frequent sequences. In ADBIS (Local Proceedings).

UK TECHNICAL ADVISORY GROUP, 2008a. UK ENVIRONMENTAL STANDARDS AND CONDITIONS (PHASE 1). UK Technical Advisory Group. Fecha de consulta: 30/08/2015

UK TECHNICAL ADVISORY GROUP, 2008b. UK ENVIRONMENTAL STANDARDS AND CONDITIONS (PHASE 2). UK Technical Advisory Group. Fecha de consulta: 30/08/2015

UNIVERSITY OF CALIFORNIA, 2013. The core of science: Relating evidence and ideas. The University of California Museum of Paleontology, Berkeley, and the Regents of the University of California.

VENKATESWARA K., GOVARDHAN A., CHALAPATI K., 2012. Spatitemporal Data Mining: Issues, Tasks and Applications. IJCSES. Vol.3. N.1

VICAIRE - Virtual campus in hydrology and water ressources, 2006. Chapter 2: WATER QUALITY CHARACTERISTICS. VICAIRE. http://echo2.epfl.ch/VICAIRE/mod_2/chapt_2/main.htm. Fecha de consulta: 30/08/2015

W3SCHOOLS, 2015. What is Google Maps?. W3SCHOOLS. http://www.w3schools.com/googleapi/. Fecha consulta: 04/10/2015.

WANG J., HAN J., 2004. BIDE: Efficient mining of frequent closed sequences. Int. Conf. on Data Eng. pp 79–90.

WORLDATLAS, 2015. United Kingdom. Worldatlas. http://www.worldatlas.com/webimage/countrys/europe/unitedkingdom/uklandst.htm#page. Fecha de consulta: 30/08/2015

YAN X., AFSHAR R., 2003. CloSpan: Mining Closed Sequential Patterns in Large Datasets. SIAM International Conference on Data Mining.

YANG Z., WANG Y., KITSUREGAWA M., 2007. LAPIN: Effective sequential pattern mining algorithms by last position induction for dense databases. Advances in Databases: Concepts, Systems and Applications. pp1020–1023.

YU S., 2012. The dynamic competitive recommendation algorithm in social network services. In Information Sciences pp 1–14.

ZAKI, M. 2001. SPADE: An efficient algorithm for mining frequent sequences. Machine Learning pp 31–60.

ZHONG N., LI Y., WU S., 2012 Effective Pattern Discovery for Text Mining. In IEEE transactions on knowledge and data engineering.